VŠB - Technická Univerzita Ostrava
Fakulta strojní
Katedra automatizační techniky a řízení

# Využití PLC pro řízení části výrobní linky

# Using PLC for Control of the Production Line

Student: Martin Štěpán
Vedoucí bakalářské práce: Ing. Jaromír Škuta, Ph.D.

Ostrava 2011

VŠB - Technická univerzita Ostrava
Fakulta strojní
Katedra automatizační techniky a řízení

# Zadání bakalářské práce

Student: **Martin Štěpán**

Studijní program: B2341 Strojírenství

Studijní obor: 3902R001 Aplikovaná informatika a řízení

Téma: Využití PLC pro řízení části výrobní linky
Using PLC for Control of the Production Line

Zásady pro vypracování:

1. Seznamte se s vývojovým prostředím pro programování PLC firmy ABB a s vývojovým prostředím pro tvorbu SCADA/MMI aplikací, systémem Control Web.
2. Seznamte se s laboratorní úlohou demonstrující část výrobní linky a připojte ji ke standardnímu rozhraní řídicího systému na bázi PLC firmy ABB.
3. Proveďte analýzu zadání algoritmu pro vybranou část řízené výrobní linky a vytvořte aplikaci pro PLC realizující Vámi vytvořený algoritmus.
4. Vytvořte aplikaci realizující konfiguraci a monitorování laboratorního modelu výrobní linky. Zohledněte možnost modulární aplikace v prostředí Control Web.
5. Zhodnoťte dosažené výsledky a navrhněte další směr řešení.

Seznam doporučené odborné literatury:

[1] BÍLÝ, R., CAGAŠ, P. & AJ. 1999. Control Web 2000. Průvodce systémem pro tvorbu a nasazení aplikací reálného času. 1. vydání. Praha: Computer Press, 1999. 382 s. ISBN 80-7226-258-0.
[2] BOYER, S. A. 1999. SCADA: Supervisory Control and Data Acquisition, 2nd Edition. New York (USA): ISA, 1999. 215 p. ISBN 1-55617-660-0.
[3] MARTINÁSKOVÁ, M. ŠMEJKAL, L. PLC a automatizace 1, základní pojmy, úvod do programování. Praha: BEN, 2002. 224 s. ISBN 80-86056-58-9.
[4] VLACH, J. Počítačová rozhraní, přenos dat a řídicí systémy. Praha, BEN-technická literatura, 1997, ISBN 80-85940-17-4.
[5] WHITT, M. D. 2003. Successful Instrumentation and Control Systems Design. New York (USA): ISA, 2003. 360 p. ISBN 1-55617-844-1.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Jaromír Škuta, Ph.D.**

Datum zadání: 17.12.2010
Datum odevzdání: 23.05.2011

VYSOKÁ ŠKOLA BÁŇSKÁ-TECHNICKÁ UNIVERZITA Fakulta strojní OSTRAVA

prof. Ing. Jiří Tůma, CSc.
*vedoucí katedry*

prof. Ing. Radim Farana, CSc.
*děkan fakulty*

Místopřísežné prohlášení studenta

Prohlašuji, že jsem celou bakalářskou práci včetně příloh vypracoval samostatně pod vedením vedoucího bakalářské práce a uvedl jsem všechny použité podklady a literaturu.

V Ostravě dne …….

……

podpis studenta

Prohlašuji, že

- jsem byl seznámen s tím, že na moji bakalářskou práci se plně vztahuje zákon č. 121\2000 Sb., autorský zákon zejména § 35 – užití díla v rámci občanských a náboženských obřadů, v rámci školních představení a užití díla školního a § 60 – školní dílo.
- beru na vědomí, že Vysoká škola báňská – Technická univerzita Ostrava (dále jen „VŠB-TUO“) má právo nevýdělečně ke své vnitřní potřebě bakalářskou práci užít (§ 35 odst. 3.).
- souhlasím s tím, že bakalářská práce bude v elektronické podobě uložena v Ústřední knihovně VŠB-TUO k nahlédnutí a jeden výtisk bude uložen u vedoucího bakalářské práce. Souhlasím s tím, že údaje o kvalifikační práci budou zveřejněny v informačním systému VŠB-TUO.
- bylo sjednáno, že s VŠB-TUO, v případě zájmu z její strany, uzavřu licenční smlouvu s oprávněním užít toto dílo v rozsahu § 12 odst. 4 autorského zákona.
- bylo sjednáno, že užít své dílo – bakalářskou práci nebo poskytnout licenci k jejímu využití mohu jen se souhlasem VŠB-TUO, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly VŠB-TUO na vytvoření díla vynaloženy (až do jejich skutečné výše).
- beru na vědomí, že odevzdáním své bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb., o vysokých a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů, bez ohledu na výsledek její obhajoby.

V Ostravě dne ……..

…….
podpis studenta

Martin Štěpán
Dlouhá 28
74101, Nový Jičín

# ANOTACE BAKALÁŘSKÉ PRÁCE

**Štěpán, M.** *Využití PLC pro řízení části výrobní linky*: *bakalářská práce*. Ostrava: Katedra automatizační techniky a řízení, Fakulta strojní VSB-Technická univerzita Ostrava, 2011, 51s. Vedoucí práce: Škuta, J.

Práce se zabývá použitím programovatelného automatu (PLC) v oblasti automatizace výrobní linky. Zabývá se nejen naprogramování celého procesu, ale i tvorbou vizualizace procesu.

Teoretická část práce ukazuje možnosti jednotlivých prostředí a to jak pro programování PLC, tak i pro programování vizualizace. V praktické části je popsána tvorba jednotlivých programů a jsou zde vysvětleny důvody pro konkrétní volby u všech postupů.

# ANNOTATION OF BACHELOR THESIS

**Štěpán, M.** *Use of PLC for operation of section of a production line: Bachelor Thesis.* Ostrava: Department of Control System And Instrumentation, Faculty of Mechanical Engineering VSB-Technical university of Ostrava, 2011, 51p. Thesis head:: Škuta, J.

In this thesis I have focused on use of programmable automatic machines (PLC) in automatization of production lines. I have focused not only on its programming but also on creation of vizualization of the whole process.

In teoretical part I started with description of programmable automatic machine (PLC), further description of backgound for visualization was made. In practical part methodology for elaboration of individual applications and clarification of reasoning for particular procedures preference were introduced.

# Obsah

# SEZNAM POUŽITÝCH ZNAČEK A SYMBOLŮ

| | |
|---|---|
| CoDeSys | Software pro programování PLC dle mezinárodní normy IEC (Controlled Development System ) |
| CPU | Centrální procesorová jednotka (Central Processing Unit) |
| DDE | Protokol definovaný firmou Microsoft pro spolupráci aplikací prostřednictvím výměny dat a povelů (Dynamic Data Exchange) |
| DHCP | Server dynamicky přidělující IP adresy (Dynamic Host Configuration Protocol) |
| EPROM | Semipermanentní typ paměti (rasable Programable Read Only Memory) |
| GND | Zemnění elektrického obvodu (GrouND) |
| HTML | Standard sjednocující formátování a zobrazování v síti internet (Hyper Text Makeup Language) |
| HTTP | Protokol umožňující komunikaci po síti internet, nebo malé firemní síti (Hyper Text Transfer Protocol) |
| IEC | Mezinárodní norma věnující se PLC, jejich funkci, provedení hardwaru a komunikací i způsobům jejich programování (International Electrotechnical Commission) |
| IP | Číslo, které jednoznačně identifikuje síťové rozhraní v počítačové síti (Internet Protokol) |
| MAC | Jedinečný indikátor síťového zařízení (Media Access Control) |
| OLE | Technologie firmy Microsoft, která umožňuje vkládání a propojování různých dokumentů a objektů (Object Linking and Embedding) |
| OPC | Standardizované komunikační rozhraní (OLE for Process Control) |
| PLC | Programovatelný logický automat (Programmable Logic Controller) |
| POU | Organizátor programů (Program Organisation Unit) |
| PROFIBUS | Standardizované sériové rozhraní určené pro technologické řídící systémy (Process Field Bus) |

| | |
|---|---|
| RAM | Paměť s libovolným přístupem(Random Access Memory) |
| RS-232 | Sériová komunikační sběrnice |
| SPS | Programovatelný logický automat (Speicherprogrammierbare Steurung ) |
| TCP/IP | Protokol pro komunikaci server-klient (Transmission Control Protocol/Internet Protocol) |
| URL | identifikátor umístění v síti internet (Universal Resource Locator) |

# 1 Úvod

Na světě existuje nespočet velkých, ale i malých, firem zabývajících se výrobou výrobků nejrůznějších funkcí, či montáží sestav různé složitosti. Tyto firmy mají společnou minimálně jednu věc, všechny se snaží vydělat co nejvíce peněz. Jednou z nejúčinnějších možností je co nejvíce zefektivnit výrobu (popřípadě montáž) a to bez použití automatizačních prostředků není možné.

Dříve se tato automatizace řešila tzv. „pevnou logikou“, jednalo se o jednotlivé logické součástky pevně napájené, nebo jinak spojené, do větších celků tak, aby byly schopny plnit aktuální požadované funkce. Ovšem postupem času se stále častěji kladl větší a větší požadavek na flexibilitu výroby z nejrůznějších technologických důvodů, doplnění o další funkce atd. Možnosti „pevné logiky“ už byly značně omezené a nenabízela rychlou a levnou změnu funkce, nebo rychlé a levné rozšíření. Tedy vznik programovatelné logiky, která by logické funkce zpracovávala stejně jako „pevná logika“, ale její změna by byla časově i finančně méně náročná, byl nevyhnutelný. Nahradily ji logiky, které požadované funkce nevykonávaly svým uspořádáním logických prvků v celek, ale plnili tyto funkce nejrůznějšími programy.

Po čase se tyto systémy vyvinuly až do podoby dnešních programovatelných logických automatů PLC (Programable Logic Controller). Při použití PLC jsou změny funkcí, rozšíření plynoucí ze změny technologie, nebo jen prosté doplnění stávající technologie, otázkou několika desítek minut a náklady na takovouto změnu jsou nesrovnatelně nižší, než tomu bylo u zastaralé „pevné logiky“. Nezanedbatelnou výhodou je také snadné připojení více jednotek mezi sebou, ale také připojení, buď jednotlivých jednotek, nebo celku, k PC, které může například shromažďovat data o výrobním procesu, nebo monitorovat výrobní nebo jiný proces v reálném čase. Získaná data jsou velmi cennou informací. Lze jimi dál zefektivňovat chod ovládaného stroje, nebo provádět korekce ovládacích pochodů.

Samozřejmě by se daly nahradit standardním PC, které nabízí mnohem větší výpočetní výkon, nesrovnatelně vyšší kapacitu paměti, přehlednější metody zobrazení průběhu řízení stroje, také by se daly vybavit speciálními kartami pro komunikaci se senzory i pro ovládání nejrůznějších akčních členů a také mnohé další. Ovšem nejsou přizpůsobené pro provoz v náročných továrních podmínkách, ať už jde o prašné prostředí, různé vnější rušení, kolísání napětí v síti, nebo odolnost proti mechanickému poškození. Avšak největší slabinou PC proti PLC je jeho možnost rozšíření o nejrůznější potřebné vstupy a výstupy. Zpravidla je zapotřebí mnohem více vstupů a výstupů (jak binárních, tak

analogových), než nám můžou tyto rozšiřující karty pro PC nabídnout, proto jsou do takovýchto podmínek nevhodné. Takovéto důvody vedou k masovému využíváni jednotek PLC v nejrůznějších odvětvích od domácího použití až po hierarchické řízení celé továrny, kompletně ovládané hotelové sítě, řízení vytápění, ovládání výtahů, řešení přístupů hostů podle čipových karet atp. V dnešní době, kdy si opravdu bez automatizace nedokážeme představit každodenní život, tvoří PLC páteř veškerého automatického řízení.

# 2 Vývojové prostředí pro PLC - CoDeSys

Vývojové prostředí CoDeSys vyvinulo mezinárodní sdružení CoDeSys Automation Alliance a představuje komplexní řešení pro programování PLC firmy ABB. Ovšem disponuje programovacím jazykem IAE, který je definován mezinárodní normou IEC 61131. Tuto normu programovacího jazyka dnes již používá drtivá většina výrobců programovatelných automatů (např.: Siemens, Mitsubishi, Unitronics, Teco atd.). Lze tedy bez ohledu na použitý hardware vytvářet ovládací programy a datové struktury.

Firma Berger Lahr jej dodává již s integrovanými knihovnami univerzálních funkcí a funkčních bloků. Tyto se u různých firem výrobců liší, a proto ho vybavují různými knihovnami programových bloků pro konkrétní výrobky. Navíc ještě pro navazující periférie poskytuje již v základu další knihovny pro zajištění potřebných funkcí, např.: Soft-Motion (který zajišťuje v součinnosti s určitým hardwarovým vybavením synchronizovaný pohyb pohonů ve více osách), PLC-open (Pro pohony, které se připojují přes standardní komunikační rozhraní), atd. [Regulační pohony, 2009]

## 2.1 Tvorba projektu v CoDeSys

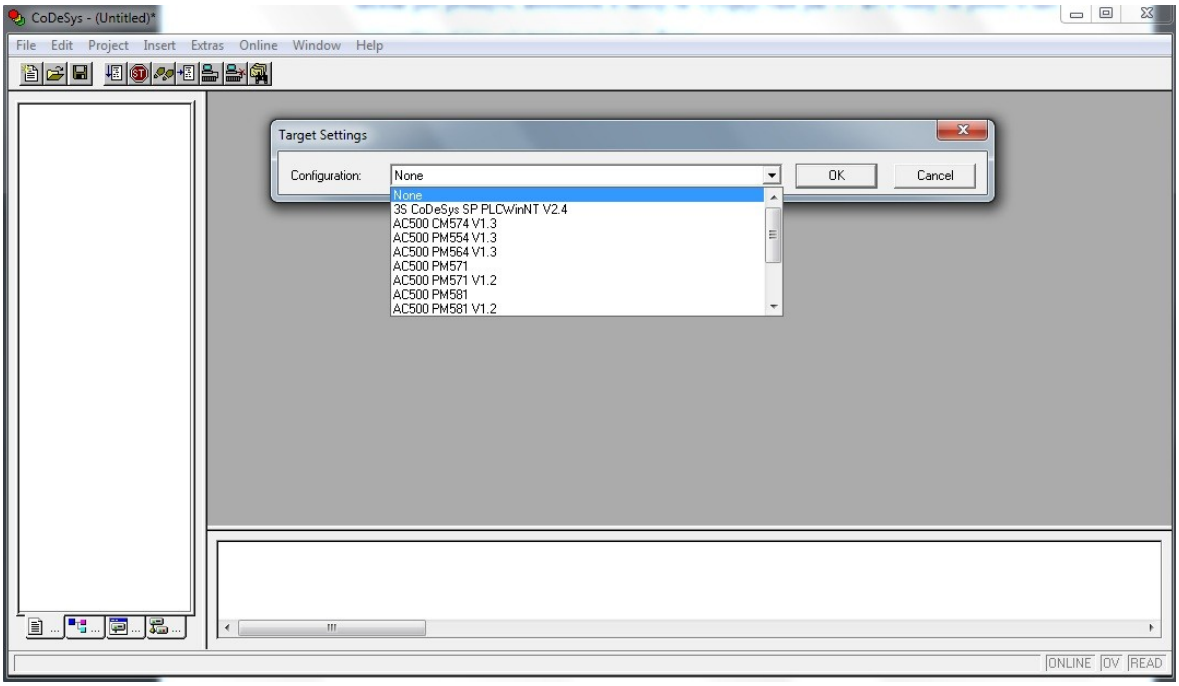
Každý vytvářený projekt je podmíněn, jak přesně definovaným zadáním, tak pečlivým zkoumáním ovládaného systému. Pomocí takto připravené úlohy je možno vytvářet program pro automat v několika jazycích, které lze v projektu libovolně kombinovat. K dispozici máme tyto jazyky:

- IL - Instruction List.
- ST - Structured Text (obdoba Pascalu).
- LD - Ladder Diagram (kontaktní plán).
- SFC - Sequential Function Chart.
- FBD - Function Block Diagram.
- CFC - Continuous Function Chart.

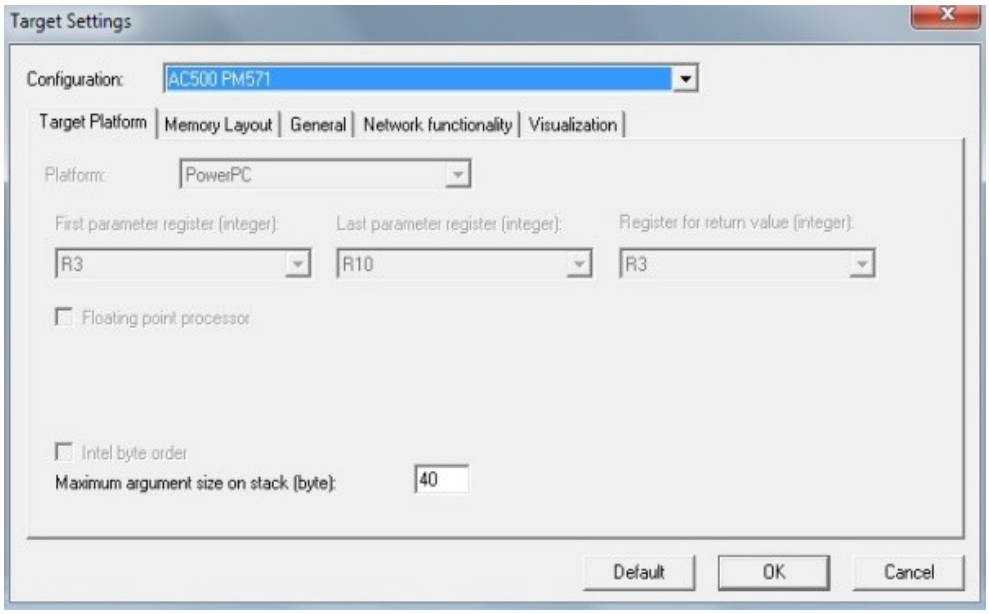
Použití CoDeSys nám umožňuje kompletně vyřešit zadaný projekt od vytvoření programu pro PLC, přes jeho odladění (pomocí reálného hardwaru, nebo v simulačním módu), testování, vizualizaci procesů, až po uvedení systému do provozu, a dokonce i vytvoření dokumentace. [Regulační pohony, 2009]

## 2.2 Grafické prostředí CoDeSys

Základ uživatelského prostředí je systém cílení tzv. Target (obr. 1). Tento systém nám umožňuje výběr konkrétního hardwarového prostředku, který můžeme snadno konfigurovat a parametrizovat (obr. 2). Poté nám software automaticky načte potřebné knihovny k tomuto hardwaru a nastaví optimální hodnoty pro komunikaci a chod aplikace a hardwarového prostředku řízení.

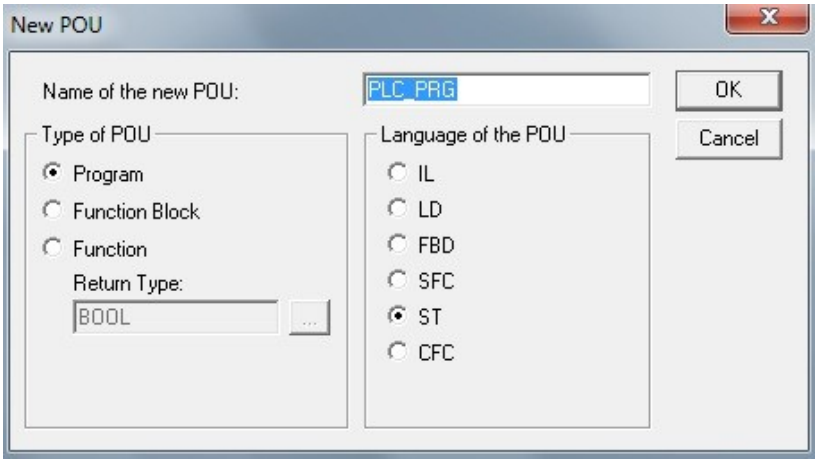



Obr. 1 *Systém cílení target*



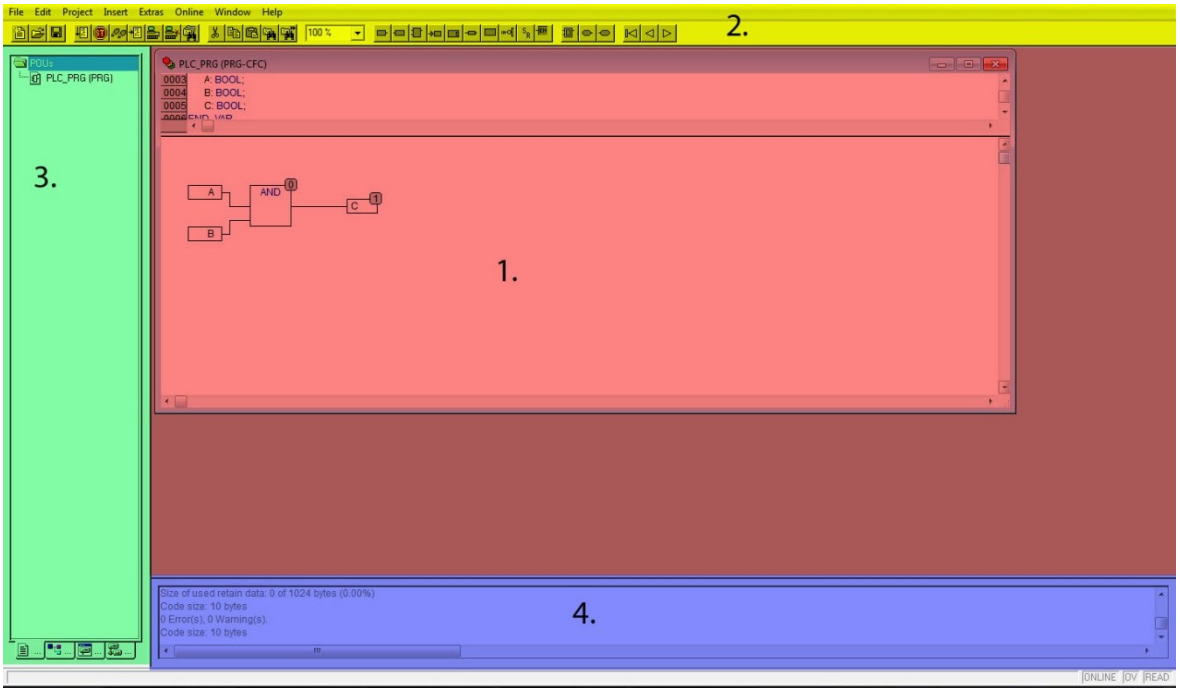

Obr. 2 *Ukázka konfigurace parametrů hardwaru*

Jakmile si nastavíme všechny potřebné parametry hardwaru, tak nás systém vyzve k volbě programovacího jazyka (obr. 3). Máme na výběr z již zmíněných šesti jazyků a dále můžeme vybírat s možností vytvoření programu, funkce, nebo jen funkčního bloku.




Obr. 3 *Volba programovacího jazyka*

Nyní se dostáváme do hlavního grafického rozhraní, kde máme pracovní plochu rozdělenou do čtyř hlavních zón, jak je vidět na obrázku (obr. 4).




Obr. 4 *Pracovní plocha CoDeSys*

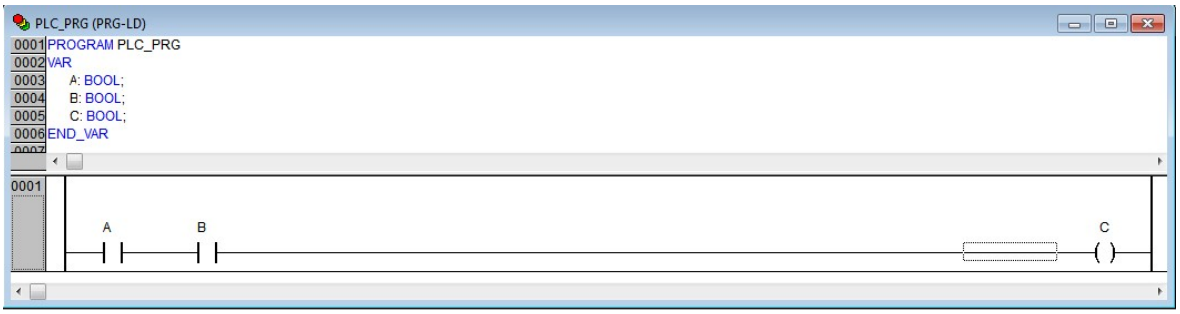
1.Hlavní pracovní prostor.

2.Panel nástrojů.

3.Objektový organizátor.

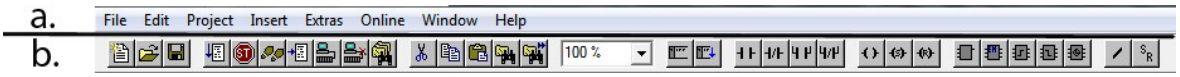
4.Okno zpráv.

***Hlavní pracovní prostor*** – slouží k vytváření samotného programu v jednotlivých programovacích jazycích a je pro většinu jazyků rozdělen na dvě části. V první části zadáváme název programu, funkce, nebo funkčního bloku a současně deklarujeme používané proměnné, popřípadě vstupní a výstupní parametry proměnných používané například pro bloky programů. V druhé části vytváříme samotný program ve formě strukturovaného textu, nebo s pomocí předdefinovaných funkcí či funkčních bloků (popřípadě funkčních schémat). Ukázku máme na obrázku (obr. 5). [CoDeSys Service Tool CST, 2010 ]



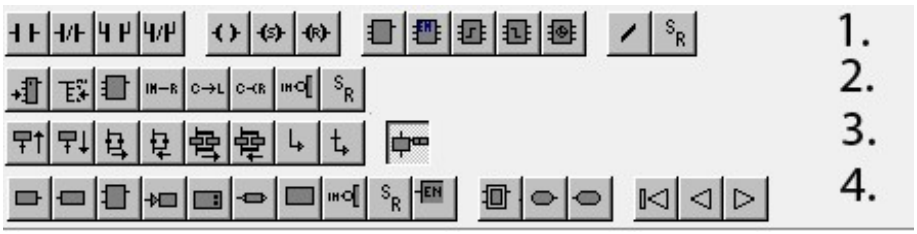

*Obr. 5* Příklad jednoduchého programu jazyka LD v pracovním prostoru

***Panel nástrojů*** – obsahuje všechny příkazy ovládající chod programu, ale i příkazy pro vytváření programu, ovládání aplikace, ukládání, načítání a mnoho dalších. Je pomyslně rozdělen do dvou částí a to část a. a část b.(obr. 6).




*Obr. 6* Panel nástrojů

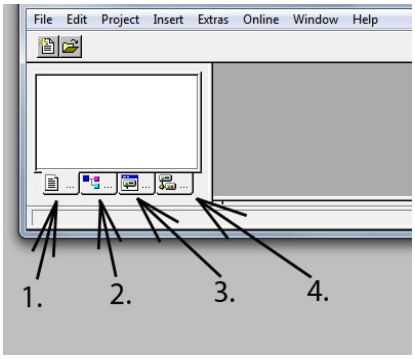
**Část a.** obsahuje veškeré přístupné příkazy zejména pro chod aplikace, uložení, načtení, editace, správce projektu, atd., zatímco v **části b**. najdeme tlačítka funkcí, z nichž jsme schopni poskládat celý program v určitém jazyce zvoleném programátorem. Takovouto lištu najdeme i v případě, že se rozhodneme programovat vizualizaci, avšak je naprosto odlišná (více v Objektovém organizátoru – vizualizace). Ukázky panelů vybraných jazyků máme na obrázku (obr. 7). [CoDeSys Service Tool CST, 2010]




1.LD.
2.FBD.
3.SFC.
4.CFC.

*Obr. 7* Panely nástrojů

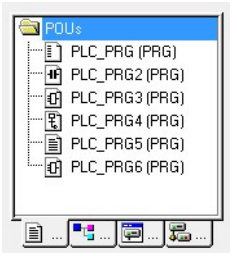
***Objektový organizátor*** – můžeme pomocí něj ovládat všechny údaje o projektu a jeho jednotlivé programy v různých, nebo i stejných jazycích, zpravujeme vizualizace, přidáváme a editujeme datové typy, nastavujeme informace o zdrojích, informace o výstupech, modifikujeme prostředky atd. Je přehledně rozdělen na čtyři části (obr. 8) obsažených v jednotlivých záložkách:




1.POUs.
2.Vizualizace.
3.Datové typy.
4.zdroje.

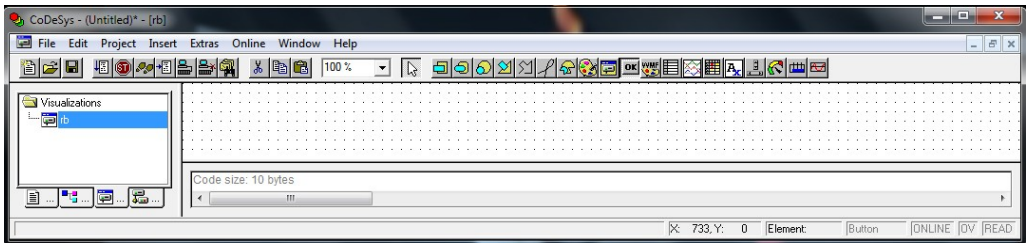
*Obr. 8* Rozdělení projektového organizátoru

První částí je tzv. ***POUs*** *(1.),* zde máme ve stromu seřazeny všechny programy projektu (obr. 9), které zde můžeme přejmenovávat, měnit pořadí a jejich typy. CoDeSys umožňuje i překládání mezi jednotlivými jazyky programování podle potřeby programátora. Jsme tedy schopni vytvořit program v jednom z jazyků a pak ho libovolně nechat přeložit do jiného.




*Obr. 9* Objektový manažer se všemi programy v jednotlivých jazycích

Další záložkou jsou ***Vizualizace*** *(2.).* V této záložce můžeme pracovat s vizualizací programovaného procesu. Můžeme vytvářet jednoduché objekty, tlačítka, indikátory, alarmy, tabulky pro trendy atd. Na obrázku (obr. 10) můžeme vidět příkazovou lištu a prostředí pro vizualizaci.




*Obr. 10* Pracovní prostředí Vizualizací

Třetí je záložka ***Datové typy*** *(3.).* Zde spravujeme uživatelsky definované datové typy, které si před svým spuštěním program alokuje v paměti počítače, na němž je spuštěn. Můžeme použít datové typy pro deklaraci čísla, textu, nebo řetězce. Přehled datových typů najdeme v tabulkách (tab. 1 a tab. 2). [CoDeSys Service Tool CST, 2010]

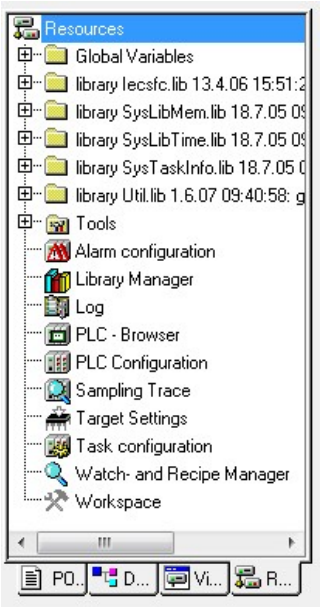
*Tab. 1 Integer Data Types (Celočíselný typ) [*CoDeSys Service Tool CST, 2010*]*

| Typ | Dolní limit | Horní limit | Použitá paměť |
|---|---|---|---|
| **BYTE** | 0 | 255 | 8 Bit |
| **WORD** | 0 | 65535 | 16 Bit |
| **DWORD** | 0 | 4294967295 | 32 Bit |
| **SINT** | -128 | 127 | 8 Bit |
| **USINT** | 0 | 255 | 8 Bit |
| **INT** | -32768 | 32767 | 16 Bit |
| **UINT** | 0 | 65535 | 16 Bit |
| **DINT** | -2147483648 | 2147483647 | 32 Bit |
| **UDINT** | 0 | 4294967295 | 32 Bit |

*Tab. 2 Ostatní typy datových proměnných [*CoDeSys Service Tool CST, 2010*]*

| Typ | Druh | Dolní limit | Horní limit |
|---|---|---|---|
| **BOOL** | Logická dvouhodnotová proměnná | 0 | 1 |
| **REAL** | Číselná proměnná | $1.175494351e-^{38}$ | $3.402823466e+^{38}$ |
| **LREAL** | Číselná proměnná | $2.2250738585072014e-^{308}$ | $1.7976931348623158e+^{308}$ |
| **STRING** | Textová proměnná | 0 (délky) | 255 (délky) |

Čtvrtou a poslední záložkou jsou ***Zdroje a nastavení*** *(4.).* Tato záložka obsahuje několik velice důležitých položek. Záložku "Tools", která umožňuje nastavit prvotní komunikaci PLC v síti. Dále "PLC Configuration" pro nastavení všech připojených modulů v systému, "Sampling trace" nám nahrazuje funkci osciloskopu pro zobrazování vstupních a výstupních hodnot za běhu programu a v neposlední řadě také "Task configuration" kde nastavujeme, nebo editujeme časování programu. [CoDeSys Service Tool CST, 2010]

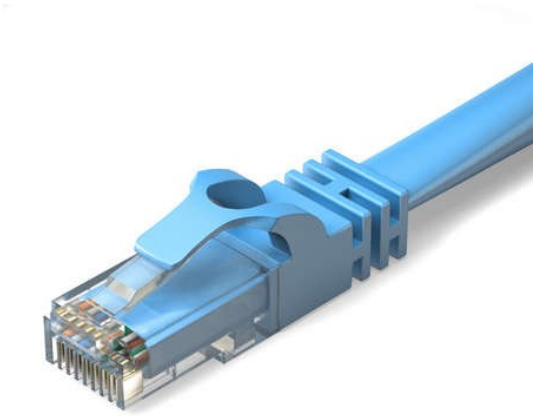



*Obr. 11* Zdroje a nastavení

***Okno zpráv*** – zde se zobrazuje přehled provedených akcí, nebo nás program varuje o chybě v programu a současně nám nabízí možné řešení. [CoDeSys Service Tool CST, 2010]

## 2.3 Komunikace s PLC

S PLC můžeme komunikovat několika různými způsoby. Ovšem pro nás bude nejvýhodnější použít stávající řešení komunikace. PLC je již připojeno do sítě a má přidělenu svou IP adresu. Přes tuto adresu je pak možné přistupovat k PLC, a tak do něj nahrávat náš program, nebo kontrolovat chod programu. Po připojení nám PLC bude posílat informace o stavu a my přímo uvidíme, který vstup je sepnutý a který rozepnutý. Tak stejně nám bude zobrazovat i stavy jednotlivých výstupů. Tato komunikace probíhá prostřednictvím zásuvky a kabelu RJ-45 (obr. 12). Norma RJ-45 má přesně definované pořadí jednotlivých vodičů a jednotlivé vodiče mají svou funkci, podle použité přenosové rychlosti. Kabel je připojen do přepínače (switche), který spojuje jednotlivé komponenty v síti a dovoluje nám komunikovat prostřednictvím IP adres. [Počítačová rozhraní, Vlach J., 2000]



*Obr. 12* Koncovka RJ-45 [www.cablestogo.com]

**IP protokol v komunikaci**

IP adresa slouží k rozlišení počítačů v sítí. V dnešní době se používají IP adresy s 32 bitovým adresováním zapsané dekadicky po jednotlivých oktetech. Například: 192.168.1.1. [Počítačová rozhraní, Vlach J., 2000]

# 3 Control Web 6.1

V dnešní době neustálých změn a optimalizací výroby, kontroly, přesnosti výroby atd. si nevystačíme s pouhými pevnými jednoúčelovými nástroji, proto vznikly víceúčelové virtuální prostředky, které jsou použitelné takřka pro všechny možné potřeby všech průmyslových odvětví. Můžeme tak vytvářet nejrůznější virtuální ovládací prvky, měřící prvky a zobrazovací prvky, jenž jsou schopny v reálném čase komunikovat s fyzickými zařízeními. Tato vizualizace nám nabízí nesmírně adaptivní prostředí, které je možno snadno a s nízkými náklady měnit podle potřeb úlohy. Jako vizualizační prostředí ovšem nebudeme používat integrované prostředí CoDeSys, ale raději použijeme nástroj orientovaný právě tímto směrem, jenž nabízí mnohem širší možnosti jak řízení, tak zobrazování procesu. Tento software vyvinula společnost Moravské přístroje, která už má v oboru průmyslového softwaru praxi více jako 16let. Možnosti této aplikace jsou obrovské a dále se rozšiřují. Vývojáři dále rozšiřují množství a typy virtuálních zařízení a dále optimalizují chod programu. [Control web 2000, 2000]

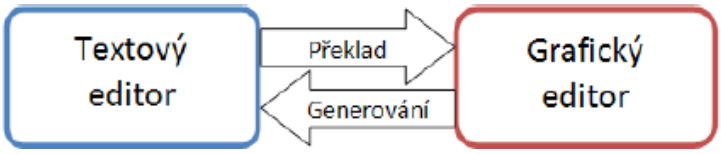
## 3.1 Tvorba projektu v Control Web 6.1

Pro vytvoření projektu musíme mít přesné podklady, nebo mít pečlivě prozkoumány všechny požadované funkce a znát možnosti připojení prostředí k úloze. Jakmile budeme mít shromážděny a nastudovány všechny potřebné materiály můžeme začít s návrhem projektu. Můžeme ho vytvářet ve dvou různých tvarech, mezi kterými lze libovolně překlápět (obr. 13). V podstatě jde o dvě vývojové prostředí s různými programovacími příkazy, ovšem změna v jednom se hned projeví i v prostředí druhém. Záleží jen na programátorovi, které prostředí si zvolí. [Manuál CW6,2008]


*Obr. 13* Vazba mezi prostředími v Control web 6.1 [*Control web 2000, 2000*]

Tato možnost překlápění je výhodná zejména pro budoucí změnu stávajícího programu. Obzvláště pokud tuto změnu provádí někdo jiný, než ten kdo ji prvotně vytvořil. Mnohdy je totiž těžké se vyznat v algoritmech, v textové i jiné formě, jiného programátora. Každý programátor má totiž jiné návyky, a proto je zejména použití grafického editoru velice výhodné a snadno použitelné.

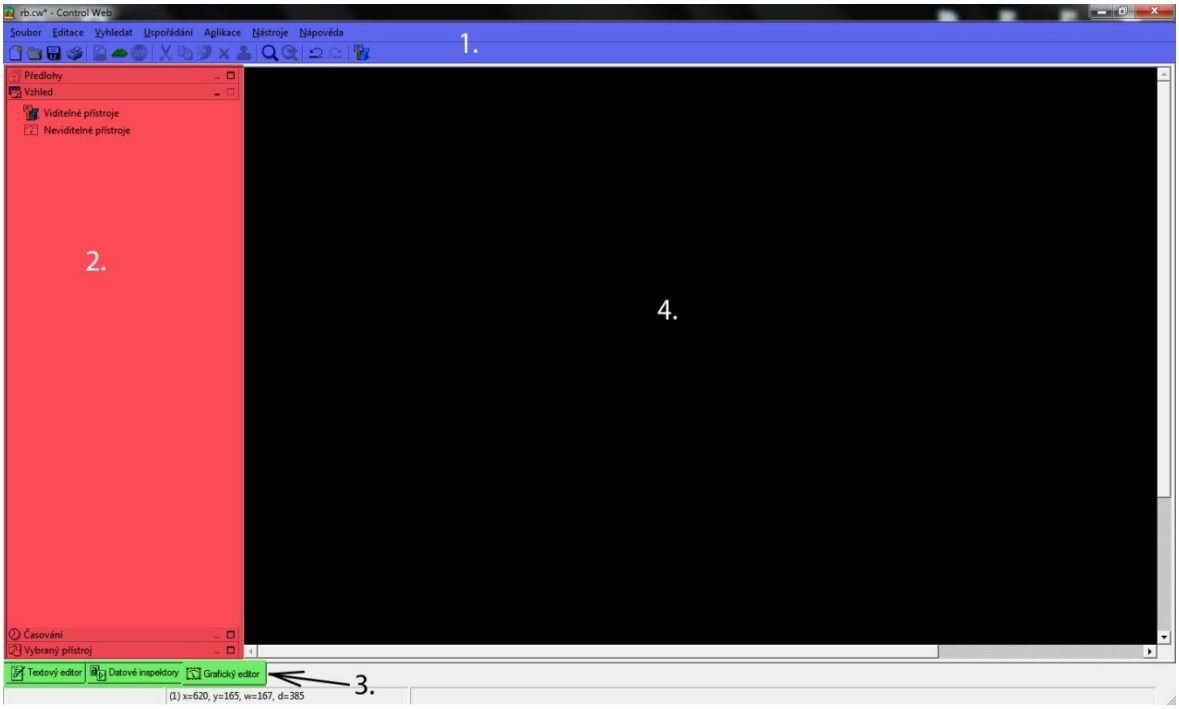
Počítač ovšem pro řízení softwaru požívá pouze textovou formu programu, proto je vždy program ukládán do této formy a grafickou formu překládá (obr. 14). Z grafické formy je pak program generován pomocí předdefinovaných algoritmů, zabudovaných přímo v programu. [Manuál CW6, 2008]




*Obr. 14* Překlad a generování v Control Web 6.1 [Control web 2000, 2000]

## 3.2 Grafické prostředí Control Web 6.1

Grafické prostředí je přehledně a i velmi intuitivně uspořádáno do čtyř hlavních částí (obr. 15). Dominantní je hlavní pracovní prostor (4.), kde realizujeme veškeré funkce, algoritmy a vzhled budoucí aplikace. [Control web 2000, 2000]




*Obr. 15* Okno grafického editoru Control Web 6.1

1. Panely nástrojů.
2. Strom možností.
3. Přepínání mezi proměnnýma a grafickým i textovým editorem.
4. Hlavní pracovní prostor.

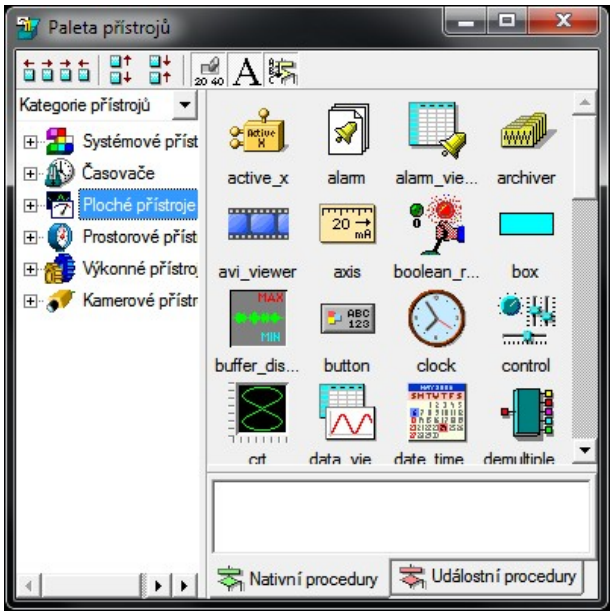
V hlavním okně, v jeho levé dolní části, dále najdeme přepínání mezi textovým grafickým editorem (3.). Při volbě grafického editoru, máme na levé straně strom (2.) s jednotlivými virtuálními přístroji, s jejich časováním, předvolbami a záložka vybraný přístroj, která shrnuje všechny nastavené elementy vybraného přístroje. Ovšem když se

rozhodneme pracovat v textovém editoru, máme k dispozici celé okno, kam vpisujeme jednotlivé algoritmy a příkazy. V horní části tohoto hlavního okna máme panely nástrojů

(1.) se standardními příkazy týkající se aplikace, ať už jejího ukládání, načítání, nebo chodu, ale zejména zde najdeme ikonu pro zobrazení okna "Palety přístrojů" (obr. 16). [Manuál CW6, 2008]


*Obr. 16* Ikona spouštění Palety přístrojů

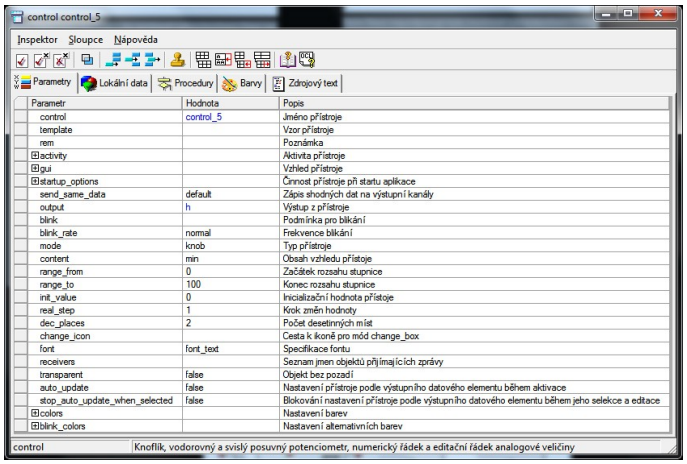
Příkaz "Palety přístrojů" nám otevře jedno z nejdůležitějších nástrojů v *Control Webu* (obr. 17). Jedná se o okno, ve kterém jsou shromážděny všechny virtuální přístroje nabízené současnou verzí 6.1. Z tohoto panelu můžeme přístroj umístit jednoduchým přetažením přístroje na pracovní plochu. Přístroje jsou zde rozděleny do několika kategorií podle jejich funkce, časování, a zda se jedná o prostorové či ploché přístroje. Přístroje umisťujeme pro přehlednost do panelů, jenž připomínají okna v klasických Windows, a tímto je i sdružujeme v další celky, kterým je možno definovat vlastnosti hromadě a dokonce je i hromadně ovládat za běhu programu. [Manuál CW6, 2008]




*Obr. 17* Paleta přístrojů

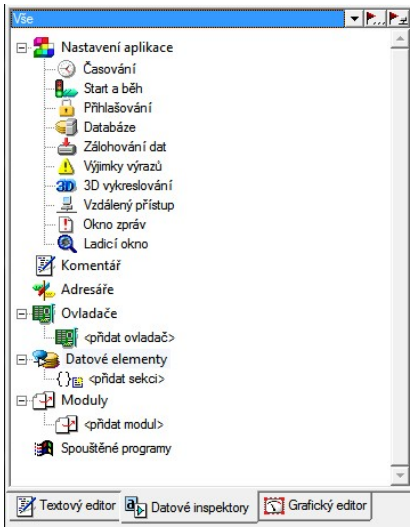
Po umístění na pracovní ploše můžeme s přístrojem dále pracovat, nastavovat velikost, umístění, barvu, vlastnosti chování a mnoho dalších. Toto vše nastavujeme v "Inspektoru přístroje" (obr. 18), který je další velmi důležitou částí grafického prostředí.

Nejenom že slouží k nastavování vlastností jednotlivých přístrojů, ale také k přidávání procedur a hlavně k navázání přístroje na proměnnou. Dále zde najdeme záložku vnitřních proměnných, v nichž jsou deklarovány pomocné proměnné, které využívá pouze tento přístroj, a zdrojového textu, kde najdeme zdrojový text pouze tohoto přístroje. Každý nástroj má po vložení nastavené veškeré hodnoty dle Control Webu jako defaultní. V okně Inspektoru přístroje najdeme po nastavení požadovaných hodnot dvě základní tlačítka, díky nimž se můžeme vrátit zpět do grafického prostředí. Jedno tlačítko nám použije změny a zavře okno Inspektoru a druhé nám zahodí změny a pouze okno zavře. [Control web 2000, 2000]




*Obr. 18* Inspektor přístroje

V hlavním okně najdeme v levé spodní části záložku "Datové inspektory" (obr. 19). Jedná se o další nedílnou součást každého projektu. V datovém inspektoru deklarujeme proměnné, přidáváme a nastavujeme ovladače pro komunikaci s vnějšími zařízeními, nastavujeme práva a přihlašování uživatelů obsluhujících aplikaci, vytváříme globální časování a mnoho dalšího. Při vytváření potřebných proměnných si můžeme vybrat ze tří druhů proměnných a to ze skaláru, pole, nebo bufferu. Po výběru základního typu si proměnnou musíme pojmenovat a zvolit její vnitřní typ podle rozsahu a hodnot vyplňovaných, nebo načítaných do proměnné. [Control web 2000, 2000]




*Obr. 19* Datové inspektory

Máme k dispozici tyto **datové typy**:

- **Boolean** – jedná se o dvouhodnotovou proměnnou (Ano-Ne, Pravda-Nepravda atd.) zabírající v paměti pouze jeden bit. Používá se pro logické funkce.
- **String** – proměnná, do které lze ukládat textové řetězce.
- **Real** – můžeme přiřadit libovolné desetinné, nebo celé číslo v rozsahu od +/-2.3× $10^{-308}$ do +/-1.7× $10^{+308}$. Hodnota tohoto typu obsadí 4 bajty paměti a jedná se o číslo s plovoucí desetinnou čárkou.
- **Shortcard** – pro nezáporná čísla od 0 do 255, která zabírají v paměti 1 bajt.
- **Cardinal** – podobná proměnná *Shortcard*, pouze s větším rozsahem od 0 do 65535. Samozřejmě zabírá více paměti (2bajty).
- **Longcard** – rozšířená proměnná typu *Cardinal* v rozsahu od 0 do 4294967295 a v paměti si alokuje 4bajty.
- **Shortint** – má za základ typ integer, jen je jeho rozsah omezen na –32 768 až 32767 a zabírá pouze 16 bitů.
- **Integer** – celočíselná proměnná, do níž jsme schopni zapsat číslo v rozsahu od –32768 do 32767 a v paměti nám bude zabírat 2 bajty.
- **Longint** – opět celočíselná proměnná, rozšířená na rozsah od –2147483648 do 2147483647 a místem v paměti velikosti 4bajty.
- **Shortreal** - pro reálná čísla v rozsahu od +/- 1.2× $10^{-38}$ do +/- 3.4× $10^{+38}$. V paměti si alokuje 4bajty.
- **Data** – jedná se o obecný datový typ, jehož vnitřní struktura se může měnit. Tato změna je definována v ovladačích, nebo přímo v přístroji.

[Control web 2000, 2000]

## 3.3 Modulární aplikace v Control Web 6.1

Systém Control Web umožňuje provozovat modulární, distribuované a synchronizované aplikace. Modulární aplikace je aplikace skládající se z více modulů. Systém Control Web modul považuje za jeden aplikační soubor, tudíž nejjednodušší aplikace je jednomodulární. Většina aplikací v systému Control Web je jednomodulární. [Manuál CW6, 2008]

## 3.4 Možnosti komunikace Control Webu 6.1 s okolím

Komunikovat s okolím jsme schopni pomocí ovladačů a další komponent. V naší aplikaci budeme používat komunikaci pouze pomocí ovladače, proto se zaměříme zejména na ně. Díky těmto ovladačům můžeme nejen komunikovat s okolím, ale i simulovat různé přístroje a jejich chování v praxi. Máme na výběr s velkého množství ovladačů, jež lze rozdělit na dvě hlavní kategorie a to na Virtuální (simulační) a Fyzické. Virtuální (simulační) ovladače se neváží na žádné skutečné kanály, ale používají se za účelem ladění aplikace. Umějí simulovat různé vstupní signály, nebo posílají signály podle definovaných parametrů. Pro nasazení aplikace v praxi se používají pouze ovladače fyzické. Mezi tyto patří zejména tyto ovladače: ASCII, TCP/IP, ADVBUF, OPC, DDE, ADVPCL atd. Pro potřeby naší aplikace se nejlépe hodí ovladač OPC, popřípadě ovladač DDE, které si popíšeme blíže. [Manuál CW6, 2008]

### 3.4.1 Ovladač OPC (OLE for Process Control)

Jedná se o standardizované komunikační rozhraní. Jedná se o neziskový standard, pomocí kterého můžeme spojit například průmyslovou automatizaci přímo s informačním systémem. Norma OPC používá ke komunikaci technologii COM, která umožňuje posílání jak jednotlivých hodnot, tak i celé pole hodnot. To jestli poslaná data jsou hodnoty, nebo pole zaleží pouze na OPC serveru, jaká data zpřístupní. Přenos dat může probíhat čtyřmi různými způsoby:

- Synchronní komunikace vždy čekající na přenos dat z/do zařízení.
- Synchronní komunikace pracující s vyrovnávací pamětí serveru (cache).
- Asynchronní komunikace (vždy komunikuje se zařízením).
- Periodická komunikace serveru se zařízením a zpětné volání klienta při změně dat.

V případě druhého a čtvrtého způsobu komunikace je zapotřebí, aby server byl schopen sám vyvolávat komunikaci s periférierni a dále informace ukládat do vyrovnávací paměti, nebo je předávat klientovi zpětným voláním. Díky používané technologii COM je každý klient schopen sám rozpoznat, s jakou verzí OPC serveru pracuje a může tímto požadovat odpovídající rozhraní dané verze. U některých klientů by mohl být problém s komunikací jednotlivých verzí, a proto každá nová v sobě integruje všechny předchozí, a tím zajišťuje kompatibilitu. [Manuál CW6, 2008]

### 3.4.2 Ovladač DDE (Dynamic Data Exchange)

Jedná se o komunikaci serveru s klientem, kdy naše aplikace je klientem přijímajícím data. V ovladači namapujeme jednotlivé kanály, které musí být shodné s kanály vysílanými serverem. Data mohou být posílány jednosměrně, ale i obousměrně po jednom kanálu. Pro zjednodušení komunikace je možné posílat data v blocích, které mohou obsahovat i data pro více kanálů. Komunikační aktivita je vyvolána ze strany klienta, nebo serveru, který otevírá komunikaci k danému tématu. Abychom zahájili správnou komunikaci, musíme znát příslušné jméno služby a jeho předmět (Topic). Tyto údaje zadáme v konfiguraci ovladače Control webu. Komunikace je realizována dynamicky, a to buď jako vyžádaná výměna dat, nebo nevyžádaná výměna dat. Jedná-li se o vyžádanou výměnu, tak jde o komunikaci vyvolanou klientem, který chce číst kanál, nebo do něj zapisovat. Naproti tomu je nevyžádaná výměna dat, kterou inicializuje server, například při změně dat. Schéma chování dynamické výměny dat můžeme vidět na obrázku (obr. 20).


*Obr. 20* dynamická výměna dat Control Web [Manuál CW6, 2008]

Ovladač DDE nám vlastně slouží jako překladač, kdy od serveru přijímá bloky dat a na druhé straně předává aplikaci pouze data v příslušném kanálu. Data s kanálu již jsou v aplikaci navázány přímo na proměnné, které jsou dále převáděny a zpracovávány podle potřeby. [Manuál CW6, 2008]

### 3.4.3 HTTP (Hyper Text Transfer Protocol) prostředí

Systém Control Web také obsahuje integrovanou komponentu, která nám umožňuje námi vytvořenou aplikaci kompilovat a dále distribuovat pomocí relativné pomalých spojení celosvětového internetu, nebo jen v rámci rychlejšího spojení v podnikové síti. Aby však bylo možno takto komunikovat a takto přijatá (v případě klienta), nebo odeslaná (v případě serveru) data byla srozumitelná bylo nutné zavést standard, který by sjednocoval veškerou komunikaci pomocí HTTP. Tento standard se jmenuje HTML(Hyper Text makeup Language), který definuje standardní způsob, jak formátovat text, zvýraznit nadpisy a jak do textu začlenit obrázky a další. Podstatnou funkcí HTML je možnost umístění do textu odkaz na jiný zdroj informací, takzvaný HYPERLINK. Takto specifikovaný HTML dokument (aplikace) je pak zpřístupněn z již zmíněných sítí. Abychom ale byli schopni se na něj připojit, je nutné mít specifikované umístění, které je definované URL (Universal Resource Locator) adresou tohoto dokumentu (aplikace). [Manuál CW6, 2008]

# 4 PLC pro ovládání výrobní linky

Jedná se o modulární PLC firmy ABB, které je možno rozšiřovat nejrůznějšími moduly, dle požadavků uživatele. Celý systém je koncipován jako stavebnice s universálními paticemi a násuvnými moduly. Navíc díky promyšlenému připojovacímu rozhraní lze vyměnit moduly bez přepojování jakékoliv kabeláže. Základem celého systému je řídicí procesor, na který dále připevňujeme jednotlivé potřebné moduly, jako jsou komunikační procesory, I/O moduly, apod. [Modulární programovatelný automat AC500, 2007]

Takto modulární systém lze použít v mnoha oblastech lidské činnosti:

- V automatizaci budov.
- Řízení výrobních strojů všeho druhu.
- Řízení energetických zdrojů (např. vodních a větrných elektráren).
- Potravinářských řízení.
- Čistíren odpadních vod.
- A mnoho dalších.

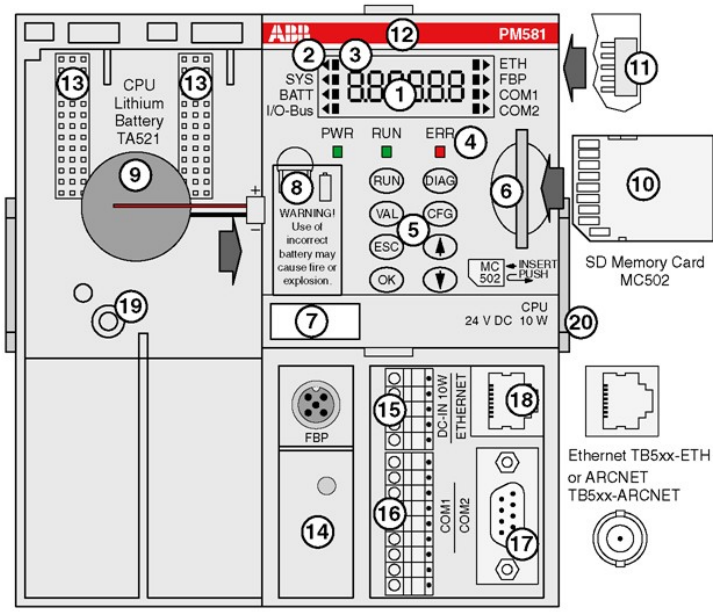
## 4.1 Procesorová jednotka PM 571

Procesorová jednotka se připevňuje jako všechny moduly na tzv. "Terminal Base", což je sběrnice, na které komunikuje s ostatními členy. Napravo od této jednotky je možno připojit až 7 digitálních nebo analogových modulů. Již základní jednotka obsahuje *Ethernet* a COM rozhraní. Dále také display pro zobrazení základních informací jako jsou: indikace připojení k síti, indikace připojení I/O modulů, stavu záložní baterie, chod programu, chybové hlášení a další. Důležitou součástí tohoto modulu je paměť pro ukládání údajů z procesu. Hlavní paměť je typu RAM (4MB) s možností zálohy baterií, další je typu Flash EPROM, nebo externě připojitelná běžná Secure Digital (SD) volitelná karta. Na čelním panelu CPU jednotky najdeme také trojici led diod, zobrazující spuštění programu, připojení napájení a chybový stav. Pro ovládání modulu zde máme osmici tlačítek, jejichž funkce je popsána v tabulce (tab. 3). [Modulární programovatelný automat AC500, 2007]

Tab. 3 Tlačítka panelu CPU jednotky

| Tlačítko | Funkce |
|---|---|
| **RUN** | Přepíná CPU do režimu Start a Stop |
| **VAL** | Slouží k procházení chybových kódů |
| **ESC** | Pro odchod z menu bez uložení |
| **OK** | Pro vstup do menu a následný odchod z menu s uložení |
| **DIAG** | Diagnostikuj a vyhodnotí chybové hlášení |
| **CFG** | Nastavuje síťovou adresu |
| ↑ | Zvyšuje hodnotu nebo nastavuje hodnotu na vyšší |
| ↓ | Snižuje hodnotu nebo nastavuje hodnotu na nižší |

Tlačítka slouží pouze pro ovládání základních funkcí, nebo základních nastavení. Všechny ostatní funkce a nastavení se provádí pomocí síťového připojení, kdy je ale potřeba tuto síť dostatečně zabezpečit proti neoprávněnému zásahu. Samotný modul totiž postrádá jakoukoliv funkci pro identifikaci připojeného uživatele, a tudíž nedokáže rozlišovat práva. [Modulární programovatelný automat AC500, 2007]




*Obr. 21* CPU jednotka [Modulární programovatelný automat AC500,2007]

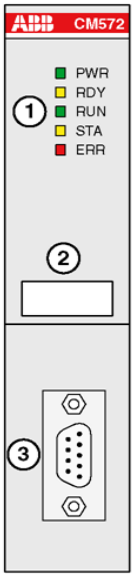
1. Sedmi segmentový display s podsvícením
2. Trojúhelníkový zobrazovač aktivních položek
3. Čtvercový zobrazovač stavu
4. Led indikující status
5. Tlačítka
6. Slot pro SD paměťovou kartu
7. Štítek
8. Místo pro lithiovou baterii
9. Lithiová baterie
10. SD paměťová karta
11. Konektor pro připojení I/O modulů
12. Slot pro CPU (CPU je integrováno)
13. Slot pro propojky (maximálně 4)
14. Rozhraní pro připojení FieldBus
15. Konektor pro připojení 24V
16. Sériový konektor COM1
17. Sériový konektor COM2
18. Síťové rozhraní (Ethernet, nebo ARCNET)
19. Otvor pro montáž na zeď
20. DIN kolejnice

## 4.2 Komunikační modul CM572

Modul CM 572 slouží ke komunikaci systému prostřednictvím sběrnice PROFIBUS DP (Process Field Bus Decentralized Periphery). Jedná se o komunikace typu master - slave, kde jde o komunikaci mezi aktivním zařízením typu "Master" a jemu přidělenými zařízeními typu "Slave". Přenosovým mediem je RS-485 kabel (standart kroucený). Čelní panel tohoto modulu je vybaven pěti indikačními LED diodami, jejichž funkce je popsána v tabulce (tab. 4). [Modulární programovatelný automat AC500, 2007]

*Tab. 4 Funkce LED indikace komunikačního modulu [Modulární programovatelný automat AC500,2007]*

| Popis LED | barva | status | Funkce |
|---|---|---|---|
| **PWR** | Zelená | ON (svítí) | Napětí je připojeno |
| | | OFF (nesvítí | Napětí není připojeno |
| **RDY** | Žlutá | ON | Modul je připraven |
| | | Cyklicky bliká | Komunikační kanál je připraven |
| | | Necyklicky bliká | Hardwarová či systémová porucha |
| | | OFF | Není připojen k zařízení |
| **RUN** | Zelená | ON | Probíhá komunikace |
| | | Cyklicky bliká | Připraven ke komunikaci |
| | | Necyklicky bliká | Chyba parametrů |
| | | OFF | Nekomunikuje |
| **STA** | Žlutá | ON | |
| | | OFF | |
| **ERR** | červen | ON | Chyba sběrnice |
| | á | OFF | Bez chyb |




1. 5 Led indikujících stav
2. Popisek
3. Komunikační rozhraní (PROFIBUS DP SUB-D, 9pinu, samička)

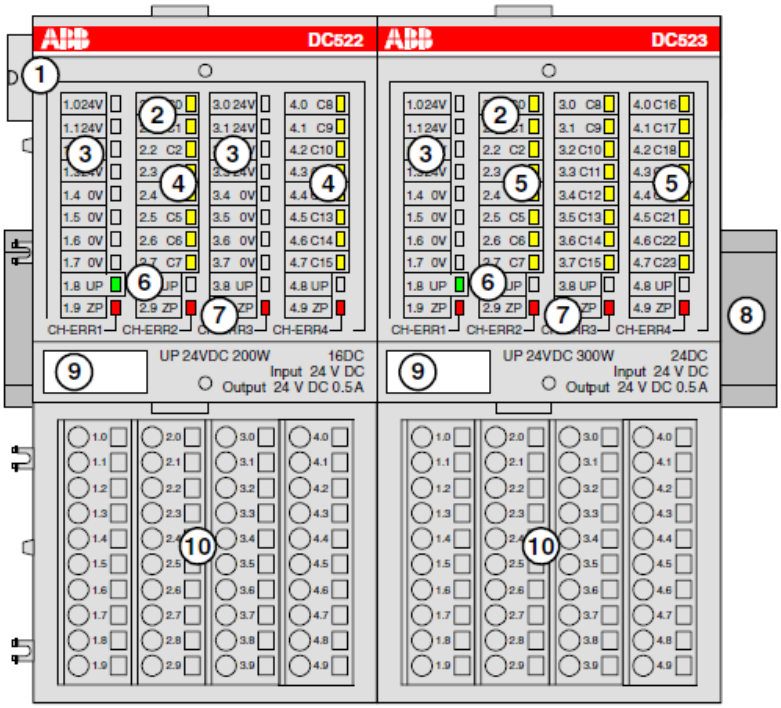
*Obr. 22* Komunikační modul CM572 [Modulární programovatelný automat AC500,2007]

## 4.3 Modul binárních vstupů/výstupů DC523

Obsahuje celkem 40 konektoru pro připojení různých zařízení. Z toho jich je 24 konfigurovatelných. Dále jsou rozděleny podle vlastností:

- 4 pro připojení senzorů, s proudem maximálně 0,5A
- 4 s nulovým výstupním napětím
- 24 binárních vstupů/výstupů.
- 4 pro připojení GND
- 4 pro připojení 24V

[Modulární programovatelný automat AC500, 2007]



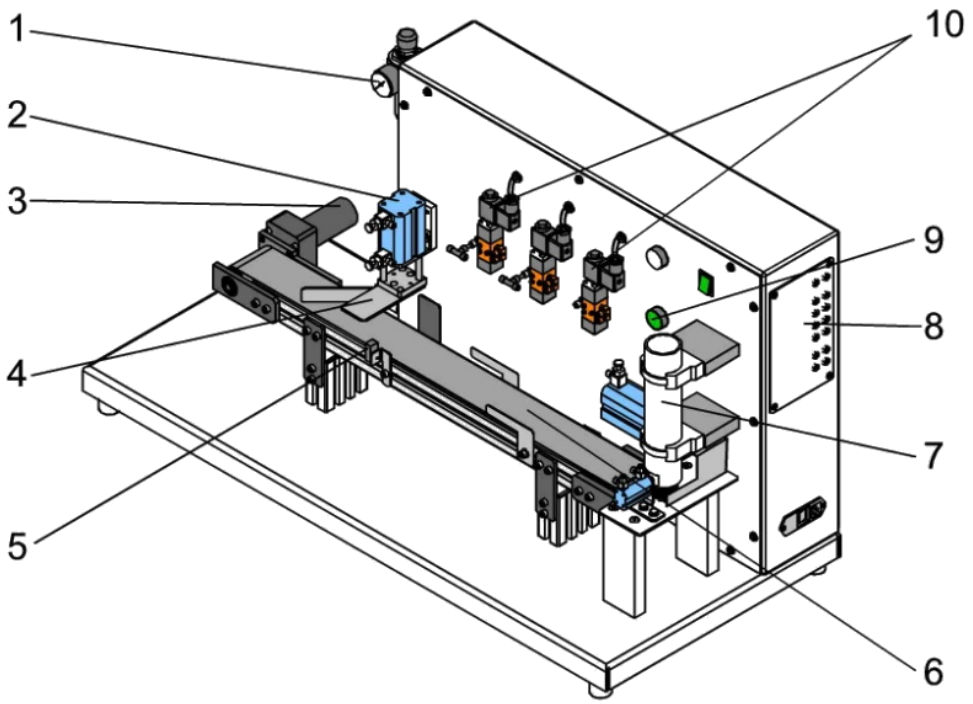

*Obr. 23* I/O modul [Modulární programovatelný automat AC500,2007]

1. Komunikační I/O rozhraní
2. Indikátory k jednotlivým I/O rozhraním
3. Indikátor připojení 24V /0,5A
4. 16 žlutých indikátorů pro vstupně/výstupní konektory C0 až C15
5. 24 žlutých indikátorů pro vstupně/výstupní konektory C0 až C23
6. 4 červené indikátory, indukující procesní napětí
7. 4červené indikátory zobrazující chyby
8. DIN kolejnice
9. Popisek
10. I/O terminál pro připojení 40 vstupně výstupních konektorů

# 5 Popis výrobní linky a vytváření algoritmu řízení

Mým úkolem bylo vytvořit algoritmus pro PLC automat, který by realizoval třídění bílých a černých výrobků pomocí modelu výrobní linky. Tento model dodala firma GUNT se sídlem v Hamburku. Firma se specializuje na výrobu a distribuci produktů pro profesní vzdělání v mnoha oblastech.

Jako první bylo zapotřebí analyzovat všechny vstupy, výstupy systému, činnosti ventilů, jejich nastavovaní a možnosti komunikace s PLC. Nejprve jsem vyrobil jednoduchou propojku mezi PLC automatem a modelem, kterou jsem využil pro tuto analýzu. Zkoumal jsem zejména chování akčních členů a snímačů, které model obsahuje (obr. 24).




*Obr. 24* Model dopravníkového pásu

1. Ukazatel tlaku vzduchu v soustavě
2. Píst ovládající akční zásah
3. Motor poháněj ící dopravníkový pás
4. Nástavec akčního zásahu, umožňující odstranění nežádoucího obrobku
5. Snímač rozpoznávající obrobky
6. Dopravníkový pás
7. Zásobník na obrobky
8. Konektory pro připojení ovládacích a čtecích zařízení
9. Indikátor přítomnost obrobku pod zásobníkem
10. Ventily A-C ovládající příslušné akční členy

Následně, abych co nejvěrněji simuloval skutečný výrobní proces, rozhodl jsem se pro doplnění modelu o operátorský panel, který by měl obsahovat základní skladbu ovládacích a kontrolních prvků. S ohledem na získané zkušenosti jsem navrhl operátorský panel. Který bude sloužit k ovládání chodu linky, dále by měl operátora informovat o stavu chodu, a také by měl obsahovat prvky pro ruční ovládání, které mohou sloužit jako diagnostický nástroj v případě poruchy, či pro případ seřízení dynamiky chodu.

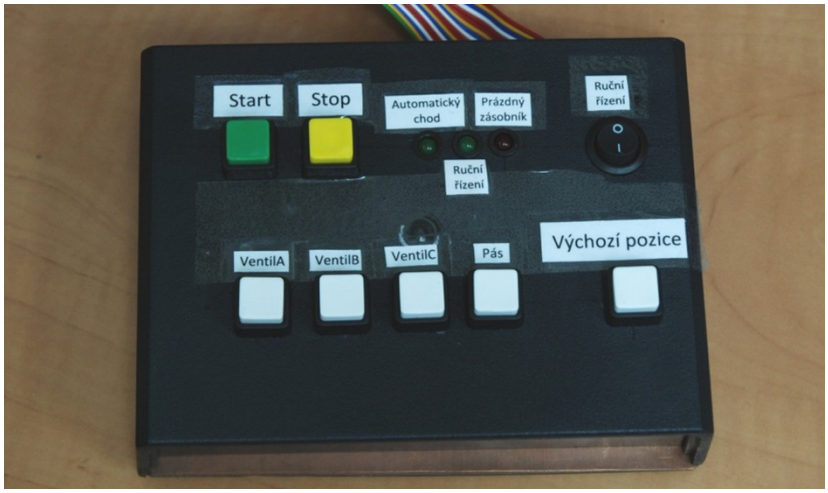
V dalším kroku bylo zapotřebí vyrobit spojení modelu, PLC a operátorského panelu. Toto jsem realizoval pomocí rozbočovače, tak aby bylo možno celý systém provozovat i bez operátorského panelu. Vyrobil jsem tedy veškeré potřebné propojovací kabely a operátorský panel, který obsahuje tlačítka:

- **Start** - pro spuštění chodu procesu.
- **Stop** - pro vypnutí činnosti procesu.
- **Ventil A** - ovládá činnost Ventilu A při ručním ovládání.
- **Ventil B** - ovládá činnost Ventilu B při ručním ovládání.
- **Ventil C** - ovládá činnost Ventilu C při ručním ovládání.
- **Pás** - spouští dopravní pás při ručním ovládání.
- **Výchozí pozice** - slouží k přípravě procesu k činnosti a to zejména k plnění zásobníku výrobky, jelikož výchozí pozice ventilu A je inverzní.
- **Ruční ovládání** - dovoluje ručně ovládat všechny akční členy systému.

a tyto indikátory:

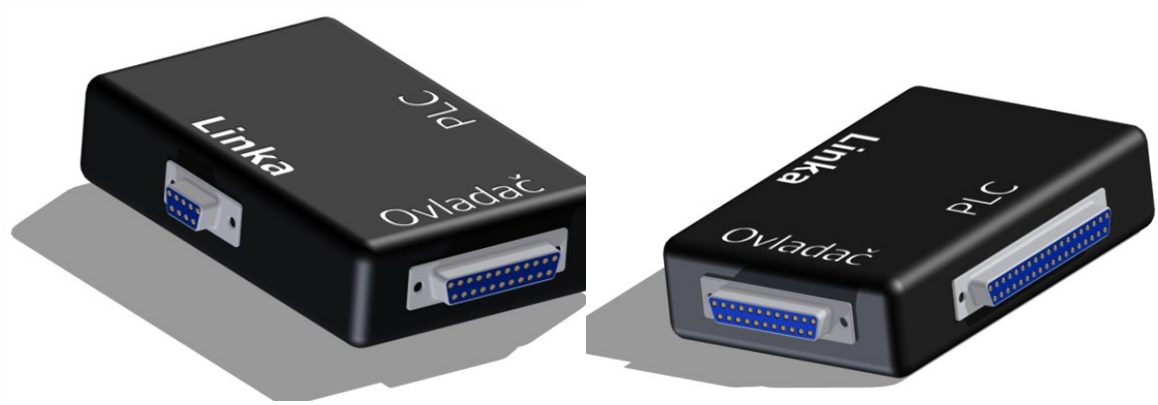
- **Došly kusy** - indikuje stav zásobníků obrobku, kdy už došly.
- **Automatický chod** - indikuje automatizovaný chod procesu.
- **Ruční řízení** - indikuje polohu přepínače ručního řízení.

Všechny tyto prvky jsou umístěny v kompaktní krabičce (obr. 25), která je pomocí 23 pinového konektoru schopna komunikovat.



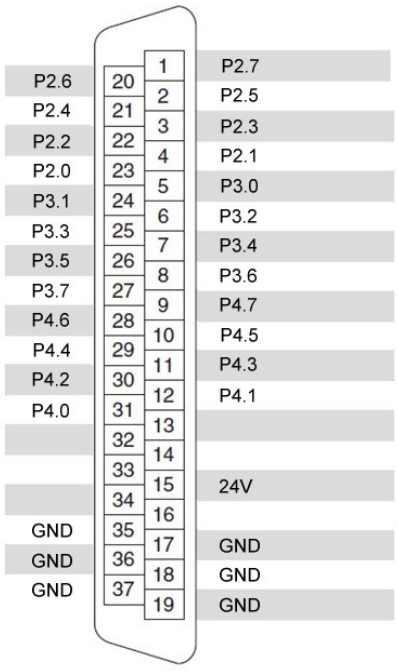

*Obr. 25* Operátorský panel

Tento rozbočovač má tři konektory pro připojení všech prvků (obr. 26). Aby nebylo možno tyto prvky prohazovat, tak jsem zvolil pro každý prvek jiný konektor (s různými počty pinů). Prvním je již zmíněný 23 pinový konektor pro připojení operátorského panelu, další je 9 pinový pro zapojení prvků na modelu a posledním je 37 pinový konektor (obr. 27), do něhož jsem zapojil binární vstupy/výstupy z PLC automatu.




*Obr. 26* Vizualizace rozbočovače

Všechny prvky, takto propojeny, jsem před spuštěním vyzkoušel pomocí multimetru, aby nedošlo ke zkratu a zároveň jsem vytvořil obrázek ukazující připojení jednotlivých pinů ke konektorům PLC, z důvodu jednoduchého přiřazování na vstupně/výstupním modulu programovatelného automatu. Abych i dalším studentům, kteří budou na tomto modelu pracovat, ulehčil práci, opatřil jsem každý koncový vodič číslem, odpovídajícím číslu portu na modulu PLC.




*Obr. 27* Zapojení 37 pinového konektoru

V dalších krocích bylo zapotřebí, abych propojil program CoDeSys s PLC automatem. Ten byl již připojen do lokální sítě, takže bylo zapotřebí, abych mu přiřadil příslušnou IP adresu, nastavil všechny moduly, které obsahuje, přidělil mu port a přihlásil se do něj. U takto fungující komunikace již je snadné nahrát vytvořený program do PLC a monitorovat jeho chod. Abych byl schopen jakkoliv komunikovat s akčními a snímacími prvky, bylo potřeba, abych namapoval proměnné na jednotlivé porty, ke kterým byly tyto členy připojeny. Přiřadil jsem tedy proměnné vstupům a výstupům a také jsem vyzkoušel jejich funkci jednoduchým přiřazováním obou hodnot (1/0) a sledoval jejich činnost. Zjistil jsem, že písty ovládané ventily A-C je zapotřebí seřídit, aby nedocházelo k rychlému vysunutí, či zasunutí pístu a tím způsobeného poškození výrobku, nebo zranění obsluhy, následkem obrobku vyhozeného z pásu. Zkoušel jsem tedy každý akční člen zvlášť a sledoval, jakým způsobem působí na obrobek. Dle pozorování jsem následně byl schopen optimálně seřídit všechny potřebné akční zásahy, pomocí škrtících ventilů, které obsahují všechny písty. Tímto krokem jsem si tedy připravil celý systém na vytváření algoritmu pro PLC, který bude řídit chování linky.

Při vytváření algoritmu sem postupoval krok po kroku, podle požadavků kladených na způsob chodu systému. Začal jsem tedy vysunutím ventilu A, který přidržuje obrobky v zásobníku a v případě potřeby je podává dále. Tuto část jsem navrhl tak, že obsluha před uvedením do chodu musí nejprve systém připravit a to stisknutím tlačítka "Výchozí pozice". Tímto dojde k vysunutí pístu A a je možno naplnit zásobník obrobky. Před dalším pokračování jsem ještě s přihlédnutím na bezpečnost vytvořil řádku v programu, která má reagovat na tlačítko "Stop", a to tak, že vše zastaví a resetuje všechny stavy pozic, aby bylo možno celý proces bezpečně zastavit a nemohlo se stát, že se obsluha zraní. Dalším krokem jsem pokračoval podáním obrobku na pracovní desku stiskem tlačítka "Start" a následným deaktivováním pístu A. Stisknutím tohoto tlačítka zároveň dojde k aktivaci automatického chodu. Na pracovní desce se nachází spínač, který po zjištění přítomnosti obrobku, aktivuje píst B. Jeho úkolem je přesunout obrobek z pracovní desky na dopravní pás. Činnost tohoto pístu je časována, aby nedošlo k situaci, že bude aktivován příliš krátkou dobu, a tím nedojde k přesunutí na pás. Naopak při příliš dlouhém časovém intervalu by mohlo dojít k spadnutí obrobku ze zásobníku, buďto na pracovní část pístu, nebo za něj a obrobek by tak vypadl z linky úplně.

Tuto část programu jsem dále doplnil o kontrolku, indikující prázdný zásobní, kdy jsem do programu vložil kontrolní smyčku, která se aktivuje při puštění pístu A a resetuje se stisknutím spínače na pracovní desce. Pokud však v zásobníku nejsou obrobky, tak nedojde ke stisknutí spínače, a tím k resetování smyčky. Ta vyčkává dvojnásobek času potřebný k umístění obrobku ze zásobníku na pracovní desku, a pokud nedojde k resetování, tak vypne všechny akční členy, rozsvítí kontrolku a uvede systém do stavu,

jako bychom připravovali systém na naplnění zásobníku. Můžeme tak doplnit zásobu obrobků a pokračovat, nebo proces ukončit.

Dojde-li k přesunutí obrobku na pás a píst B se začne vracet do výchozí pozice, je třeba spustit chod pásu. Toto sepnutí je synchronizováno s pohybem pístu B, tak aby byl pohyb obrobku co nejplynulejší a nedocházelo k jeho zbytečnému opotřebování.

Nyní už se obrobek nachází na dopravním pásu a ten ho dopravuje až k senzoru, který musí rozpoznávat barvu obrobků. Tento senzor vlastně rozpozná jen to, že obrobek není černý. V případě že projede černý obrobek, tak ho senzor vůbec nezaznamená. S touto zkušeností bylo zapotřebí pracovat při dalším návrhu programu. Změřil jsem tedy maximální čas potřebný k přejetí celého dopravního pásu a ten jsem použil v dalším cyklu, který je aktivován spuštěním pásu a deaktivován dvěma různými způsoby. První je dovršení maximální doby cesty obrobku po pásu. To znamená, že senzor obrobek nedetekoval a je tedy černé barvy. Druhý způsob je detekování obrobku jiné než černé barvy, ale to spustí ještě podcyklus, který má za úkol ovládat píst C, a tím dostat obrobek jiné barvy do postraní bedýnky. Po skončení všech těchto operací se deaktivují všechny cykly a to je signál pro nový začátek celého procesu od začátku. Systém tak může pracovat plně automatizovaně. Tento automatizovaný chod je dále indikován rozsvícením diody na operátorském panelu.

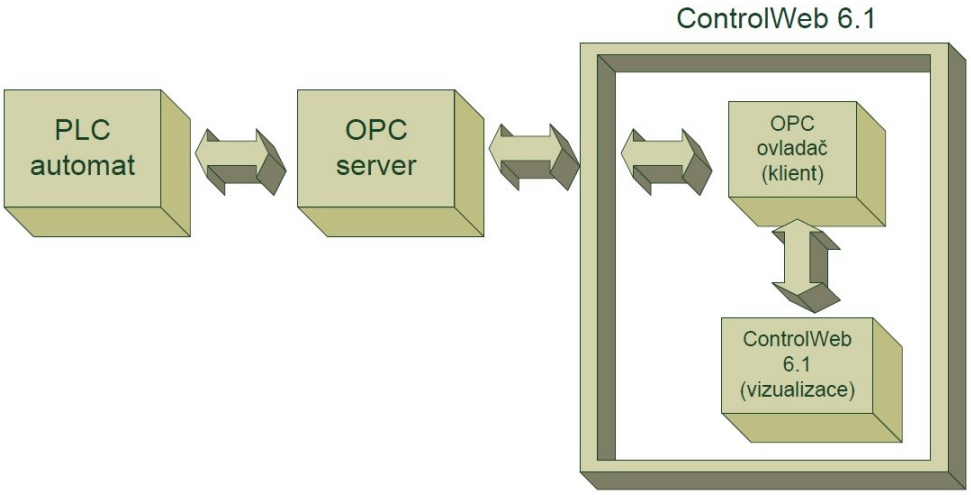
Dále jsem program opatřil dominantním vypnutím systému v případě stisknutí tlačítka *Stop,* které je v inverzním stavu obsaženo v celém programu jako nutná podmínka chodu. Jak už jsem zmínil dříve, dalším prvkem systému je možnost ručního ovládání systému jednotlivými tlačítky. Aby nemohlo dojít k jakémukoliv ohrožení nebo poškození, není možné prvky ručně ovládat, při zapnutém automatickém chodu. Z tohoto důvodu je panel opatřen přepínačem pro ruční řízení, který chod vypne a uvede do výchozích pozic jednotlivé členy. V mém případě se jedná o běžný přepínač, ale v reálném provozu by to bylo realizováno například přepínačem se zámkem, aby mohla tuto činnost provádět pouze pověřená osoba. Po přepnutí tohoto spínače je tedy možno ovládat všechny prvky jednotlivě. Pro opětovný návrat k automatickému chodu stačí přepnout zpět přepínač ručního řízení a začít celý proces od začátku.

Chod celého systému jsem po dokončení hrubého algoritmu testoval a zjišťoval drobné nedostatky v časování a nastavení polohy snímače. Všechny tyto poznatky jsem využil k vytvoření konečného algoritmu. Zkoušel jsem i různé působení poruch a chybových stavů a snažil se udělat systém odolný vůči všem těmto poruchám. Vytvořil jsem také vývojový diagram logické struktury programu. Logické schéma je obsaženi v příloze k této práci.

# 6 Propojení pomocí OPC serveru

Jakmile byl funkční a optimální celý proces, potřeboval jsem vytvořit komunikaci, která by byla schopna výměny dat s později vytvořenou vizualizací. Na mnou použitém PLC automatu byl již zprovozněn OPC server a byl připojen do školní sítě. Využil jsem tedy tuto komunikaci. Pro konfiguraci OPC serveru jsem použil nástroj, který je součástí programového vybavení programu CoDeSys. Ale nejprve bylo potřeba v otevřeném projektu s programem pro PLC nastavit potřebné proměnné, které se budou odesílat přes OPC komunikaci. To jsem provedl v záložce nastavení, kde se nachází položka Workspace. Zde jsem nakonfiguroval proměnné, které budu potřebovat, a také jim přiřadil konkrétní vlastnosti, jako je například možnost zapisování do ní z externího přístupu. Díky těmto vlastnostem budu moci proces ovládat i pomocí vizualizace. Aby Byly informace nahrány do PLC, tak jsem pomocí již zmíněného nástroje CoDeSys OPC konfigurátor provedl potřebou konfiguraci PLC. Musel jsem správně nastavit všechny IP adresy a hlavně název přiřazené aplikace, pak už jsem jen nechal tuto konfiguraci nahrát do PLC. Těmito kroky jsem nastavil OPC server a bylo možno z něj číst a posílat na něj data.

Byl také zapotřebí další nástroj pro správu OPC serveru, ale tentokrát na straně klienta. K tomuto jsem použil taktéž integrovaný nástroj aplikace ControlWeb. Jedná se o nástroj Konfigurace OPC ovladače. V tomto nástroji jsem spustil vyhledání OPC serveru, ten pak vyhledal všechny včetně toho mého a ten jsem použil pro konfiguraci. Zde jsem přiřadil jednotlivým proměnným kanály, na kterých budou dostupné. Jakmile jsem nastavil vše potřebné, mohl jsem pomocí této utility vygenerovat mapovací a parametrická soubor, který jsem dále použil pro nastavení ovladače v samotné vizualizaci.

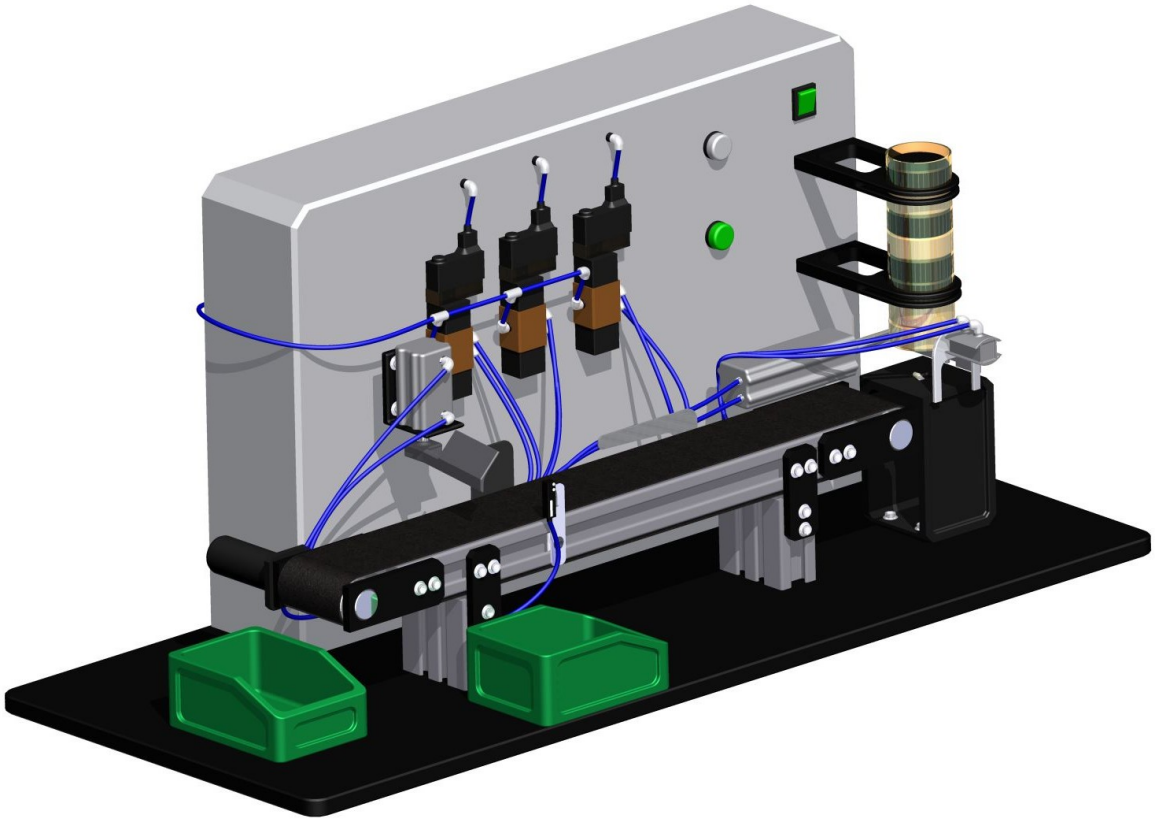



*Obr. 28* OPC komunikace PLC s vizualizací

# 7 Zhotovování a připojení vizualizace procesu

Dalším krokem bylo vytvoření vizualizace, která by měla kopírovat chod linky a měla by také umožňovat ovládání chodu linky.

Nejprve jsem si zhotovil prostorový model celé linky (obr. 29), včetně všech potřebných akčních a indikujících členů. Model jsem modeloval pomocí programu Autodesk Inventor 2011, kde jsem použil reálné osvětlení, simulující osvětlení použité na skutečném modelu. Abych dosáhl co nejvěrnější vizualizace, použil jsem renderování obrázků, které je integrováno v již výše zmíněném programu. Pomocí této renderace jsem vytvořil nejen pozadí celého systému vizualizace, ale také nejrůznější menší komponenty, které jsem později použil pro zvýšení realističnosti vizualizace, jako svrchní či naopak spodní vrstvy.

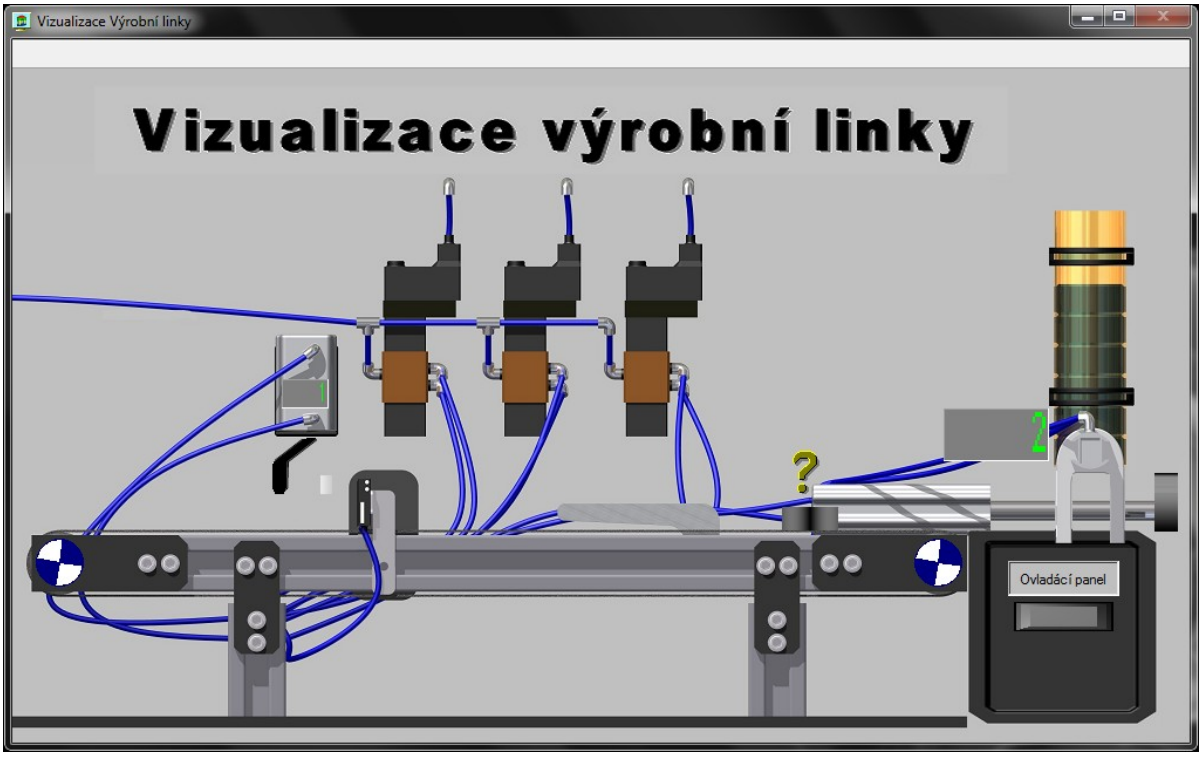


*Obr. 29* Prostorový model dopravního systému

Ve vizualizačním prostředí programu ControlWeb jsem si připravil z těchto obrázků základ, poskládáním jednotlivých vrstev, tak aby věrně kopírovaly chod skutečné soustavy. Do takto připraveného základu jsem ještě potřeboval vložit objekt, kterým bych simuloval pohyb obrobku. Pohyb lze v tomto prostředí simulovat jedině pomocí

komponenty MultiLabel, která zobrazuje soubory typu ICO. Obrobek jsem vytvořil pomocí programu iconworkshop, tak aby jej napodoboval a uložil ho jako soubor ICO, který jsem použil jako zobrazení v prvku MultiLabel. Taktéž jsem realizoval i pohyb akčních členů a pístu, který má za úkol přesouvat obrobek z podložky na dopravníkový pás. Dále jsem umístil další objekty simulující rotaci válců, které realizují pohyb pásu a také animaci akčního zásahu. Pohyb pístu přidržujícího obrobky v zásobníku jsem neanimoval, jelikož z pohledu vizualizace není viditelný.

Protože nemáme informaci o tom, jaký výrobek se nám nachází na pásu, do doby než projede skrze čidlo, rozhodl jsem se ho do doby průjezdu čidlem zobrazovat jako černý a se symbolem otazníku, simulující nejasnost ve skutečné barvě obrobku. Stav jednotlivých ventilů, ovládajících akční členy, se zobrazuje jako rozsvícená dioda v případě aktivního stavu. Umístil jsem zde i informativní indikátor počtu neshodných obrobků, jako by byl integrován do pístu akčního zásahu. Dalším indikátorem, kterým jsem vybavil vizualizaci, je celkový počet kusů, jenž je umístěn poblíž zásobníku, odkud jsou podávány obrobky, aby byl co nejpřehlednější a intuitivně rozpoznatelný. Díky jejich umístění jsem nemusel ani jeden indikátor vybavit popiskem.



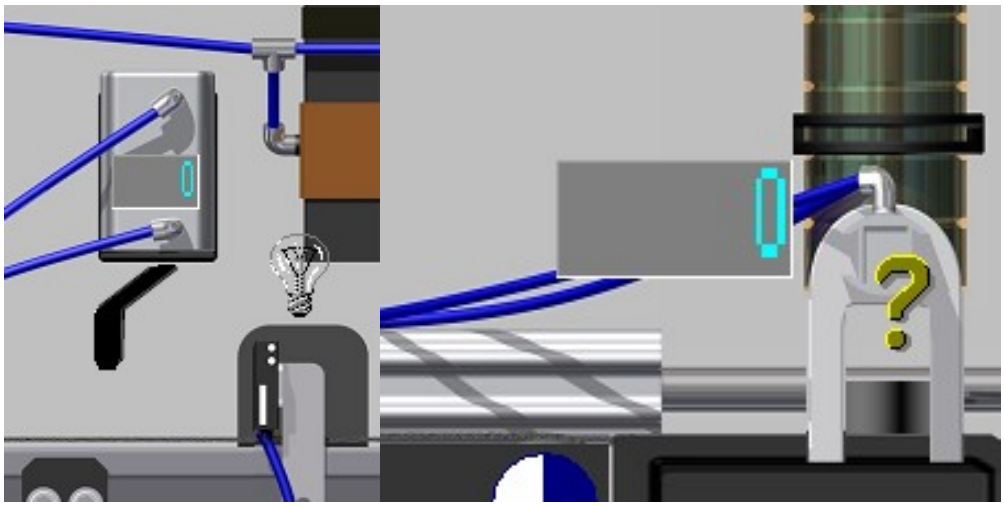

*Obr. 30* Vzhled vizualizace

Abych byl schopen přiřadit proměnné jednotlivým kanálů, tak bylo zapotřebí použít vygenerovaný mapovací a parametrický soubor, u něhož jsem si přiřadil jednotlivým kanálů svá jména proměnných, se kterými budu pracovat v rámci vizualizačního prostředí.

S přihlédnutím na budoucí rozšiřování či úpravu vizualizace jsem proměnné pojmenoval stejnými jmény jako v běžícím programu v PLC automatu.

Nyní jsem postupně doplňoval program tak, aby při určitém spuštění na reálné lince realizoval stejný pohyb ve vizualizaci. Začal jsem stejně jako při vytváření programu pro PLC. Tedy nejprve animací funkce pístu A, který spouští obrobky ze zásobníku. Toto je pouze indikováno rozsvícením kontrolního světla na těle ventilu. Dalším krokem bylo čekání na signál ze spínače umístěného na pracovní desce. Jakmile se sepne, tak se má animovat přesouvání obrobku na pás a ten se zapne. Toto jsem realizoval pomocí nástroje "program", do kterého jsem vepsal sled několika operací, které zajišťují stejný chod jako v reálu. Souběžně s tímto sem vytvořil další program, který bude ovládat chod obrobku. Tento program jsem vytvořil tak, aby byl schopen simulovat chod jak černého tak i bílého obrobku. Jelikož musí zabezpečit to, že v případě bílého obrobku je nutné jej odstranit již dříve z linky, a tím je čas běhu kratší. Poslední animovanou část jsem vytvořil simulaci pohybu akčního zásahu. Na toto jsem opět vytvořil zvlášť komponentu "program", do které jsem vepsal algoritmus zabezpečující správný pohyb. Nyní jsem odladil všechny tyto programy, aby dynamika a čas pohybu jednotlivých členů odpovídaly realitě. Pro časovou synchronizaci jsem použil vlastnosti komponenty program, která umožňuje nastavit různě rychlé časování.

V této fázi jsem vizualizaci dovybavil funkcemi dvou počítadel, které zobrazovaly celkový počet kusů a počet bílých kusů (obr. 31). Tyto dva algoritmy jsem opět vytvořil pomocí komponenty program, kterou jsem oproti ostatním časoval pomaleji, aby nedocházelo ke zbytečnému zatěžování PC, na němž program poběží.



*Obr. 31* umístění počítadel

Z tohoto pohledu jsem i každou funkci nějakého členu vytvořil v samostatném programu. To umožňuje vizualizaci pracovat na více jádrech procesoru moderních PC a s tím spojenou dostatečnou časovou rezervu pro zpracování instrukcí. A také dalo možnost

přesného odladění rychlosti chodu členů, díky možnosti rozdílného časování těchto programů.

Dále jsem vytvořil další panel (obr. 32), který se otevírá pomocí tlačítka na hlavním panelu a obsahuje obdobu hardwarového ovladače umístěného u modelu. Nejprve jsem si připravil obrázek na pozadí a poté nad něj umístil menší obrázky, které obsahují všechny ovládací prvky. Použil jsem tuto variantu, abych se neodchyloval od skutečného vzhledu a rozmístění prvků. Pro přiřazování do proměnných jsem v tomto případě použil dvě z možných procedur u komponenty *image*. První procedurou bylo spuštění vyvolané stisknutím tlačítka myši, které nastavuje příslušnou hodnotu do kanálu, a druhou procedurou bylo uvolnění tlačítka myši, jenž ji vrací zpět do původního stavu.

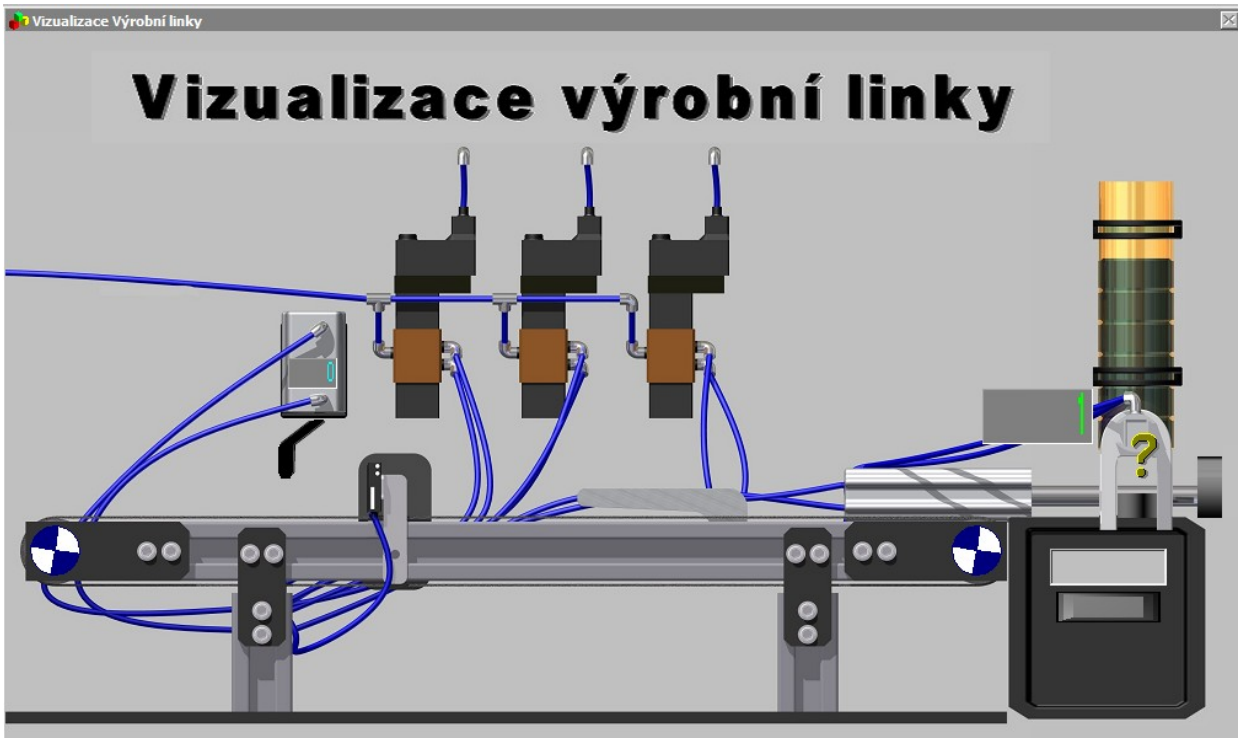

*Obr. 32* Ovládací panel ve vizualizaci

Nadefinoval jsem jednotlivé tlačítka a akční členy přesně podle chování skutečného ovladače. Je tedy možné celou linku ovládat na dálku.

Aby nemohlo dojít k neoprávněnému zásahu, doplnil jsem práva uživatelů a to na tlačítko zpřístupňující ovládací panel. Ostatní komponenty jsou zpřístupněny všem, jelikož u těchto komponent nehrozí žádné riziko ohrožení, nebo poškození.

Vytvořenou vizualizaci jsem chtěl také zpřístupnit pomocí sítě internet (obr. 33). Aplikace ControlWeb má pro toto integrovanou komponentu, kterou jsem využil. Při vytváření webového rozhraní jsem nastavil název aplikace, barvu pozadí a možnosti ovládání a zobrazování. Ovládací panel jsem přes toto rozhraní zakázal úplně, aby nemohlo dojít k neoprávněné manipulaci s linkou, tudíž je možno pouze sledovat chod linky. Jelikož zde není možná komunikace v reálném čase, je zde použita, na konci

zdrojového kódu vygenerovaného aplikací ControlWeb, funkce obnovovaní. Použil jsem obnovování po co možná nejkratším čase, pro co nejrychlejší odezvu zobrazení aktuálního stavu chodu linky.




*Obr. 33* Webové rozhraní vizualizace

# 8 Závěr

V této bakalářské práci jsem se nejprve seznámil s prostředím programu CoDeSys, který slouží k tvorbě programu, s nímž pracuje PLC firmy ABB. Také jsem prozkoumal veškeré možnosti komunikace tohoto programu s PLC automatem a zvolil ten nejvhodnější, aby bylo možno nahrát ovládací program do paměti CPU jednotky.

Následně jsem nastudoval možnosti programu ControlWeb pro tvorbu vizualizací, které je možno napojit na reálnou úlohu a tuto nejen monitorovat, ale je zde možnost i ovládání dané úlohy. Tento program obsahuje také možnost zabezpečení ve formě různých úrovní s přiřazenými právy. Dále jsem porovnal možnosti komunikace tohoto programu s připojenou úlohou a z dostupných variant jsem zvolil již realizované řešení ve formě OPC komunikačního protokolu.

Po seznámení se se softwary jsem vyhledal informace a specifikace kompletu PLC jednotky, kterou jsem měl za úkol připojit na dopravníkový pás a ovládat jej. Jako doplněk linky jsem ji osadil operátorským panelem, obsahující prvky pro ovládání a indikaci. Po prozkoumání vlastností všech modulů PLC jsem zhotovil rozbočovač, který obsahuje konektory pro propojení všech hardwarových komponent.

V dalších krocích jsem zrealizoval kontrolu funkčnosti komunikace PC (programu CoDeSys) s PLC. Následně jsem analyzoval chování celého systému, abych byl schopen vytvořit program, který bude správně ovládat chod modelu. Během vyhodnocování analýzy jsem postupně vytvářel ovládací program s hlavní prioritou bezpečnosti obsluhy. Dalším cílem byl bezproblémový chod i v krizových či poruchových stavech.

V části vizualizační jsem začal tvorbou prostorového modelu pomocí programu AutoCad Inventor. Tento model jsem použil pro tvorbu pozadí a popředí základní plochy ve vizualizaci. Následně jsem vytvořil pomocné programy, které simulují chod modelu. V tomto programu jsem kladl důraz na co nejmenší náročnost na hardware PC, na kterém poběží vizualizace. Tuto skutečnost jsem realizoval pomocí co nejméně přístrojů, které jsou absolutně časovány a pracovaly by, i když by program zdánlivě nic nevykonával, a mohl by způsobit, že by PC nebyl schopen vykonávat všechny příkazy v definovaném časovém intervalu.

Následně jsem všechny již vytvořené části spojil v jeden celek, který spolu komunikuje pomocí OPC serveru. Linku tedy můžeme ovládat jak pomocí hardwarového ovladače, tak i prostřednictvím vizualizační aplikace.

Celý proces je vytvořen co nejefektivněji s ohledem na současné možnosti snímačů a indikátorů. Dále by bylo možno celý systém doplnit dalšími snímači a celý proces ještě zefektivnit a také zabezpečit, nebo jej doplnit o webovou kameru, aby bylo možno ve vizualizačním softwaru kontrolovat i vizuálně stav procesu.

# Seznam použité literatury

BÍLÝ, Radek, et al. *Control web 2000 : Průvodce systémem pro tvorbu a nasazení aplikací reálného času*. Praha : Computer press, 1999. 382 s, ISBN 80-7226-258-0.

BOYER, S.A. *SCADA: Supervisory Control and Data Acquisition*. 2nd Edition. New York (USA) : ISA, 1999. 215 s. ISBN 1-55617-660-0.

BUCHTA, Richard. Modulární programovatelný automat AC500. *Automa*. 2007, 03, s. 1. Dostupný také z WWW: <www.odbornecasopisy.cz>.

*CoDeSys Service Tool CST* [online]. 2008 [cit. 2010-10-07]. Dostupné z WWW: <http://www.3s-software.com>.

CONTROL WEB 6. *Manuál.* Alcor - Moravské přístroje, a.s. [online]. [cit. 2009-15-11]. Dostupný z WWW: <http://www.mii.cz/>.

FOŘT, Petr; KLETEČKA, Jaroslav. *Autodesk Inventor : funkční navrhování v průmyslové praxi* . 2. aktualizované vydání. Brno : Computer Press, 2007. 318 s. ISBN 978-80-251-1773-6.

*G.U.N.T.* [online]. 2010 [cit. 2010-09-27]. Auomation. Dostupné z WWW: <http://www.gunt.de>.

Internet Protocol. In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, 19. 10. 2007, last modified on 26. 11. 2010 [cit. 2010-12-04]. Dostupné z WWW: <http://cs.wikipedia.org/wiki/Internet_Protocol>.

MARTINEK, Radislav. *Senzory v průmyslové praxi* . 1.vydání. Praha : BEN, 2004. 199 s. ISBN 80-7300-114-4.

Profibus. In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, [cit. 2010-10-14]. Dostupné z WWW: <http://cs.wikipedia.org/wiki/Profibus>.

*Regulační pohony* [online]. 2009 [cit. 2010-09-27]. Programy. Dostupné z WWW: <http://www.regulacni-pohony.cz.

ŠMEJKAL, Ladislav. *PLC a automatizace : Sekvenční logické systémy a základy fuzzy logiky*. Praha : BEN-technická literatura, 2009. 208 s, ISBN 80-7300-087-3.

ŠMEJKAL, Ladislav; MARTINÁKOVÁ, Marie. *PLC a automatizace : Základní pojmy, úvod do programování*. Praha : BEN-technická literatura, 2009. 224 s, ISBN 978-80-86056-58-6.

ŠOFER, P*. Využití SCADA/MMI systému pro podporu laboratorních měření*. Ostrava: VŠB-TUO, kat. ATŘ- 352, 2008. 47 s. Bakalářská práce.

VLACH, Jaroslav. *Počítačová rozhraní : Přenos dat a řídící systémy*. 2.rozšířené vydání. Praha : BEN, 2000. 176 s. ISBN 80-7300-010-5.

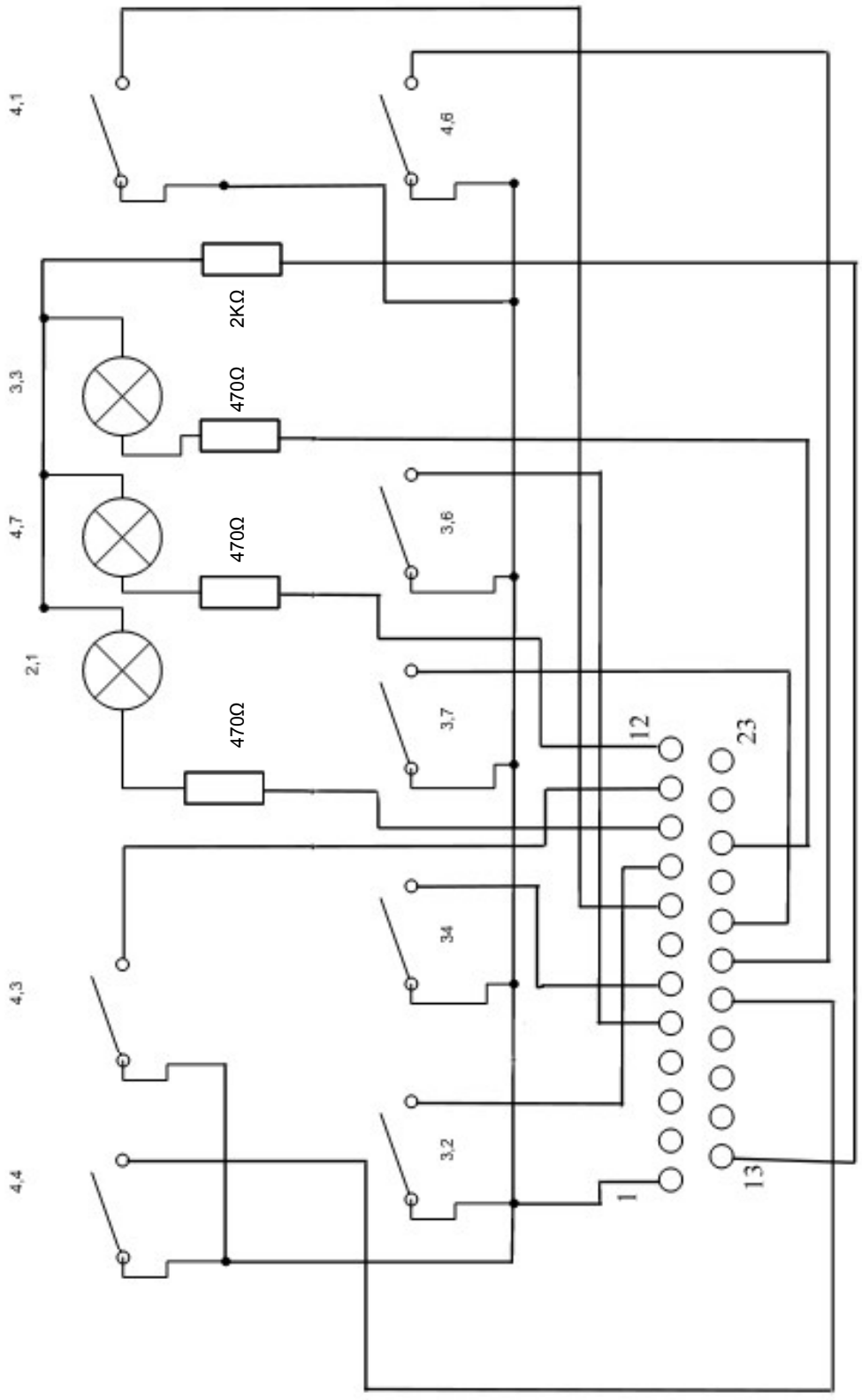
VLACH, Jaroslav. *Řízení a Vizualizace technologických procesů*. Praha : BEN, 1999. 160 s. ISBN 80-86056-66-X.

WHITT, M.D. *Successful Instrumentation and Control Systems Design*. New York (USA) : ISA, 2003. 360 s. ISBN 1-55617-844-1.

# Přílohy

# Seznam příloh

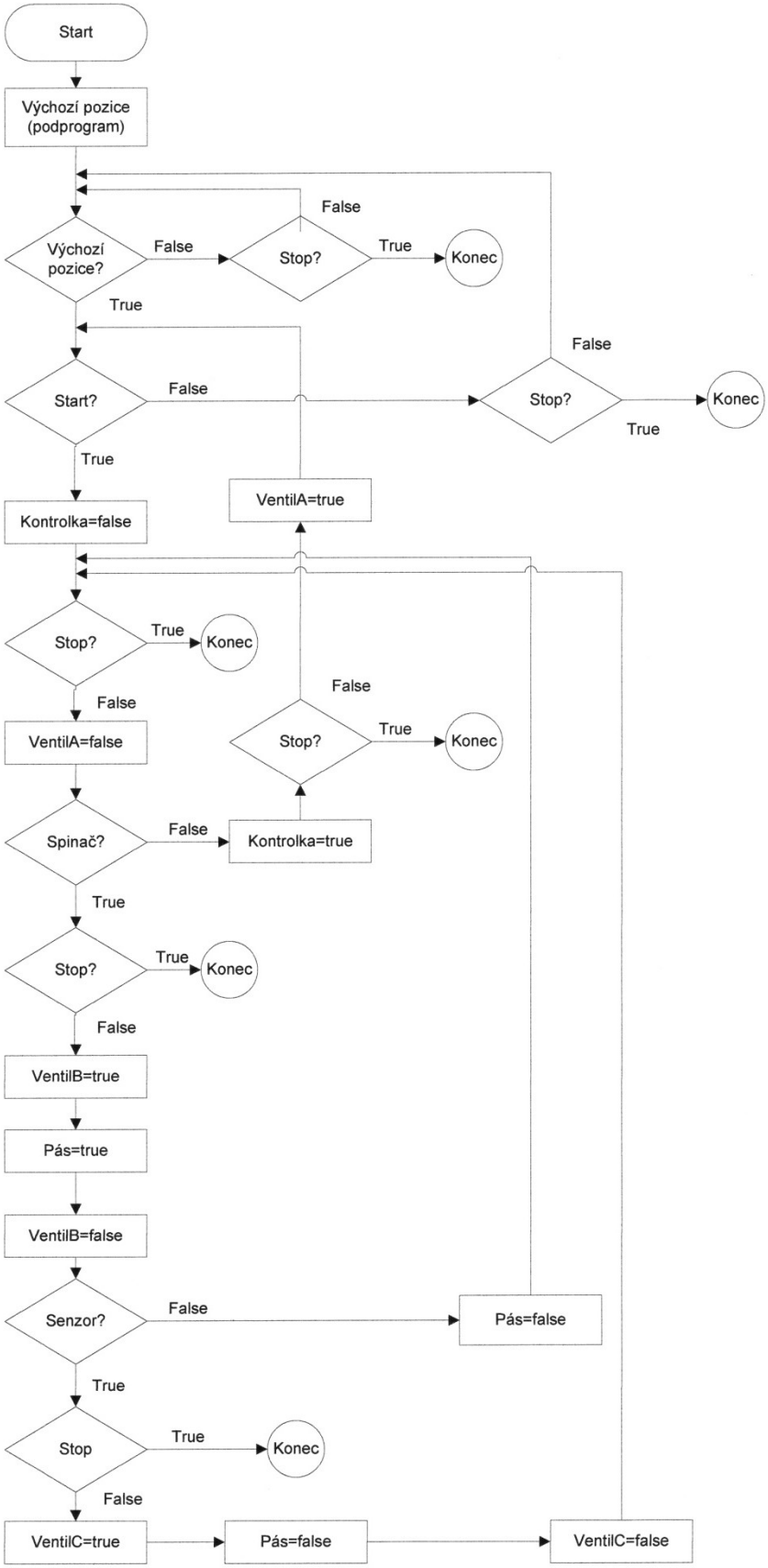
# I. Elektrické schéma zapojení ovladače výrobní linky

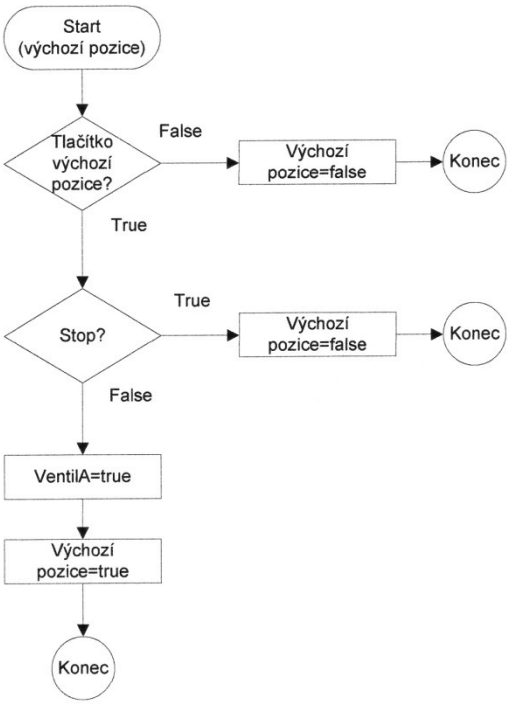


# II. Propojení konektorů s PLC modulem

| 37 pin | 23 pin | 9 pin | PLC | Koncové zařízení |
|---|---|---|---|---|
| 1 | | 1 | 2.7 | Ovládání Ventilu B |
| 2 | | 2 | 2.5 | Mechanický spínač |
| 3 | | 3 | 2.3 | Optický snímač |
| 4 | 10 | | 2.1 | Kontrolka automatického chodu |
| 5 | 7 | | 3.0 | |
| 6 | 9 | | 3.2 | TlačítkoVentilA |
| 7 | 6 | | 3.4 | TlačítkoVentilB |
| 8 | 5 | | 3.6 | Tlačítko Pás |
| 9 | 12 | | 4.7 | Kontrolka ručního řízení |
| 10 | 11 | | 4.5 | Tlačítko Stop |
| 11 | 4 | | 4.3 | |
| 12 | 8 | | 4.1 | Přepínač ručního řízení |
| 13 | | | | |
| 14 | | | | |
| 15 | 13 | 4 | 24V | |
| 16 | | | | |
| 17 | | | GND | |
| 18 | | | GND | |
| 19 | 1 | 5 | GND | |
| 20 | | 7 | 2.6 | Ovládání pásu |
| 21 | | 8 | 2.4 | Ovládání Ventilu A |
| 22 | | 9 | 2.2 | Ovládání Ventilu C |
| 23 | 23 | | 2.0 | |
| 24 | 22 | | 3.1 | |
| 25 | 21 | | 3.3 | Kontrolka prázdného zásobníku |
| 26 | 20 | | 3.5 | |
| 27 | 19 | | 3.7 | TlačítkoVentilC |
| 28 | 18 | | 4.6 | Tlačítko Výchozí pozice |
| 29 | 17 | | 4.4 | Tlačítko Start |
| 30 | 16 | | 4.2 | |
| 31 | 15 | | 4.0 | |
| 32 | | | | |
| 33 | | | | |
| 34 | | | | |
| 35 | | | GND | |
| 36 | | | GND | |
| 37 | | | GND | |

# III. Logické schéma programu pro PLC




Start
(výchozí pozice)
Tlačítko
výchozí
pozice?
False
Výchozí
pozice=false
Konec
True
Stop?
True
Výchozí
pozice=false
Konec
False
VentilA=true
Výchozí
pozice=true
Konec